© May 2026 | IJIRT | Volume 12 Issue 12 | ISSN: 2349-6002

Smart Traffic Violation Detection with Edge AI Optimization

Dr. V. Manoranjithem[1], M. Sivakumar[2], M. Kalaiyarasan[3], V. Surya[4]
[1]*Associate Professor, Department of Computer Science,*
Kalasalingam Academy of Research and Education, Krishnankoil
[2,3,4]*III B.Sc CS, Department of Computer Science,*
Kalasalingam Academy of Research and Education, Krishnankoil

***Abstract*—This paper presents an intelligent Edge AI-based traffic violation detection system that automates vehicle speed monitoring, violation evidence captures, and real-time alert generation using computer vision and deep learning techniques. The proposed system processes uploaded video footage through a sequential pipeline of eight integrated modules: video input processing, vehicle detection using TensorFlow-based YOLO architecture, vehicle tracking with unique ID assignment, speed prediction through frame-by-frame displacement analysis, configurable threshold management, automated evidence image capture, email notification generation, and violation record database management. By leveraging OpenCV for video frame extraction and TensorFlow for deep learning-based object detection, the system accurately identifies vehicles and calculates their real-world speed using pixel-to-distance calibration techniques. When a vehicle exceeds the predefined speed threshold, the system automatically captures a timestamped image of the violating vehicle, overlays violation data including vehicle ID, speed, time, and sends an immediate email alert with attached evidence to the designated traffic in-charge. The Edge AI optimization ensures low-latency processing by performing computation locally without cloud dependency, making it suitable for real-time traffic enforcement applications. Experimental results demonstrate that the system achieves reliable vehicle detection accuracy, precise speed estimation, and instantaneous violation notification, thereby addressing the limitations of existing manual monitoring systems that lack automated evidence capture and real-time alert mechanisms. This solution provides traffic authorities with an efficient, scalable, and legally-valid tool for automated traffic law enforcement and improved road safety.**

***Index Terms*— Edge AI, Traffic Violation Detection, Vehicle Speed Estimation, Deep Learning, OpenCV, Real-Time Alert System**

I. INTRODUCTION

The exponential growth in urban vehicle density has led to increased traffic violations, particularly overspeeding, which remains a leading cause of road accidents and fatalities worldwide. Traditional traffic enforcement methods rely heavily on manual monitoring by personnel, radar guns, or fixed speed cameras that are labor-intensive, costly to maintain, and incapable of continuous 24/7 surveillance across multiple lanes simultaneously. Furthermore, existing automated systems often operate as isolated units that capture violations without intelligent integration, lacking the capability to process video footage in real-time, generate tamper-proof digital evidence with vehicle identification, or instantly notify authorities for immediate action. The absence of automated evidence capture mechanisms results in legal challenges during citation issuance, while delayed violation processing reduces the deterrent effect on potential offenders. These limitations necessitate an intelligent, scalable solution that can leverage modern computer vision and artificial intelligence technologies to transform conventional traffic monitoring into a proactive, automated enforcement system.

This paper proposes an Edge AI-optimized traffic violation detection system that addresses these challenges by integrating deep learning-based vehicle detection, real-time speed estimation, automated evidence generation, and instant notification mechanisms into a unified framework. The system utilizes Python, OpenCV for video processing, and TensorFlow for implementing YOLO-based vehicle detection models to identify and track vehicles across video frames with unique identifiers. By employing pixel-to-distance calibration and frame rate analysis, the system accurately predicts vehicle speed and compares it against configurable thresholds. Upon detecting a violation, it automatically captures and timestamps vehicle images, overlays violation data, and triggers email notifications to traffic authorities with complete evidentiary information. The Edge AI architecture ensures low-latency processing by performing all computations locally, eliminating dependency on cloud infrastructure and enabling deployment at roadside checkpoints, traffic signals, or highway surveillance points. This comprehensive approach provides traffic enforcement agencies with an efficient, legally-valid tool for automated violation detection, real-time alerting, and systematic record maintenance, ultimately contributing to enhanced road safety and traffic rule compliance.

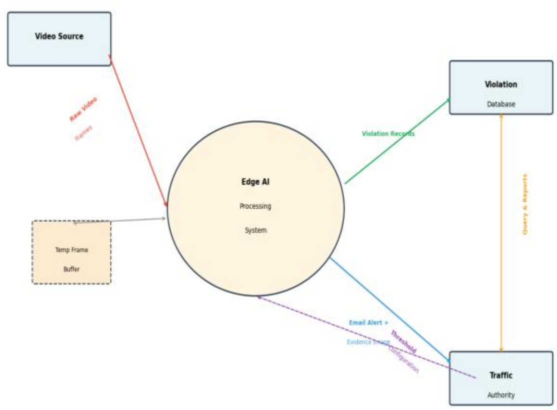


Fig 1: Architecture Diagram

II. IDENTIFY, RESEARCH AND COLLECT DATA

Video Input Module

The Video Input Module serves as the entry point for the system, handling the acquisition and preprocessing of video footage from various sources including uploaded files, CCTV streams, or recorded surveillance videos. This module utilizes OpenCV's VideoCapture functionality to extract frames sequentially while maintaining critical metadata such as frame rate (FPS), total frame count, video resolution, and timestamp information for each frame. The module implements buffer management to handle high-resolution video streams efficiently, ensuring smooth frame delivery to subsequent processing stages without memory overflow. Additionally, it performs initial validation checks on video format compatibility and integrity, rejecting corrupted files and standardizing input parameters for consistent downstream processing. The extracted frames are timestamped and passed to the vehicle detection module along with frame sequence numbers that are essential for accurate speed calculation through temporal displacement analysis.

Vehicle Detection Module

The Vehicle Detection Module employs a TensorFlow-implemented YOLO (You Only Look Once) deep learning architecture to identify and localize vehicles within each video frame with high precision. The module loads a pre-trained model weights file optimized for vehicle detection across multiple categories including cars, motorcycles, buses, and trucks, achieving robust performance under varying lighting conditions and camera angles. For each input frame, the detection algorithm performs a single forward pass through the convolutional neural network, generating bounding box coordinates, confidence scores, and class labels for all detected vehicles. The module implements non-maximum suppression to eliminate redundant overlapping detections and filters out low-confidence predictions below a configurable threshold. The output of this module consists of structured detection data including bounding box dimensions, centroid coordinates, vehicle classification, and confidence metrics, which are essential inputs for the subsequent tracking module to establish vehicle identities across consecutive frames.

Vehicle Tracking Module

The Vehicle Tracking Module assigns and maintains unique identifiers for each detected vehicle across sequential frames using centroid tracking algorithms combined with Kalman filtering for motion prediction. The module compares bounding box centroids between consecutive frames, calculating Euclidean distances and intersection-over-union (IoU) metrics to associate current detections with existing tracked objects. When a new vehicle enters the frame, the algorithm generates a unique vehicle ID and initializes a tracking record containing its detection history, movement path, and temporal information. For vehicles that exit the frame or become temporarily occluded, the module maintains their state for a configurable grace period before deregistration, ensuring continuous tracking through brief遮挡 scenarios. The tracking data, including centroid displacement vectors and frame-by-frame position logs, is continuously updated and fed into the speed prediction module for accurate velocity calculations.

Speed Prediction Module

The Speed Prediction Module calculates real-world vehicle velocity by analyzing the displacement of tracked vehicles across multiple frames using calibrated pixel-to-distance conversion factors. The algorithm first establishes a reference calibration by mapping pixel distances to actual road measurements using known reference points or lane width standards within the video frame. For each tracked vehicle, the module computes the Euclidean distance traveled in pixels between consecutive frames and converts this to real-world distance using the established calibration factor. By incorporating the video frame rate (FPS) and the time interval between frames, the system calculates instantaneous speed in kilometers per hour using the formula: speed = (distance meters / time seconds) × 3.6. The module implements moving average filtering across multiple frame pairs to smooth out instantaneous fluctuations and provide stable speed estimates, while also detecting acceleration and deceleration patterns for comprehensive violation analysis.

Threshold Management Module

The Threshold Management Module provides configurable speed limit parameters and violation decision logic that continuously compares detected vehicle speeds against predefined thresholds. This module allows traffic authorities to set different speed limits for various vehicle types, road segments, or time periods through a simple configuration interface, accommodating diverse enforcement requirements. For each tracked vehicle, the algorithm maintains a violation status flag that triggers when the calculated speed exceeds the threshold by a configurable margin, accounting for sensor tolerance and natural speed variations. The module implements hysteresis to prevent repeated violation triggers from borderline cases, requiring sustained over speeding over multiple frames before confirmation. When a violation is confirmed, the module immediately signals the evidence capture module with the vehicle ID, exact violation timestamp, measured speed, and excess margin, while vehicles operating within limits continue normal processing without triggering further actions.

Evidence Capture Module

The Evidence Capture Module automatically generates forensic-quality violation evidence by extracting and annotating the exact video frame where overspeeding was confirmed. Upon receiving a violation trigger from the threshold module, the algorithm retrieves the corresponding frame from the video buffer and overlays critical information including the vehicle's bounding box, unique vehicle ID, measured speed in km/h, violation timestamp with millisecond precision, and the speed threshold value. The module applies OpenCV drawing functions to create permanent visual annotations on the image, ensuring the evidence clearly displays all violation parameters within a single frame for legal admissibility. Multiple evidence frames may be captured to show the vehicle's position before, during, and after the violation moment, providing comprehensive documentation. The annotated images are saved in standard formats with filenames encoding the vehicle ID, date, and time, and their file paths are passed to both the notification module for immediate alerting and the database module for permanent record storage.

Notification Module

The Notification Module implements an automated email alert system that instantly communicates violation details to designated traffic authorities using SMTP protocol integration. Upon receiving evidence image paths and violation data, the module constructs a formatted email message containing the vehicle ID, measured speed, threshold limit, violation time and date, and location identifier if configured. The algorithm attaches the captured evidence image(s) to the email and sends it to pre-configured recipient addresses using secure authentication with Gmail or organizational mail servers. The module implements retry logic with exponential backoff to handle temporary network failures, ensuring reliable notification delivery. Additionally, the system maintains a notification log tracking sent alerts, delivery status, and timestamps for audit purposes, while also supporting optional SMS or push notification extensions for critical violations requiring immediate attention.

Database Management Module
The Database Management Module provides persistent storage and retrieval capabilities for all violation records, vehicle tracking data, and system configuration parameters using SQLite or MySQL database integration. For each confirmed violation, the module stores a comprehensive record including vehicle ID, measured speed, threshold value, timestamp, evidence image file path, and notification status. The database schema supports complex queries for generating violation reports by date range, vehicle type, or speed severity, enabling traffic authorities to analyze patterns and identify repeat offenders. The module also maintains a vehicle tracking history log that stores movement trajectories and speed profiles even for non-violating vehicles, supporting traffic flow analysis and future system optimization. Regular database backup mechanisms ensure data integrity and recovery capabilities, while indexed queries maintain fast access speeds even as the violation record database grows over time.

Edge AI Optimization Module
The Edge AI Optimization Module ensures efficient system performance on resource-constrained hardware by implementing model compression, inference optimization, and computational load balancing techniques. The module converts trained TensorFlow models to TensorFlow Lite format, reducing model size by approximately 75% while maintaining detection accuracy within acceptable limits for traffic enforcement applications. Quantization techniques convert 32-bit floating-point operations to 8-bit integer computations, significantly reducing memory bandwidth and accelerating inference on edge devices like Raspberry Pi or NVIDIA Jetson platforms. The module implements frame sampling strategies that process every nth frame for detection while using lightweight optical flow for intermediate frame tracking, balancing accuracy with computational efficiency. This optimization enables real-time processing of 30 FPS video streams on edge hardware without cloud offloading, ensuring low-latency violation detection suitable for deployment at remote traffic monitoring locations.

Calibration and Configuration Module
The Calibration and Configuration Module provide tools for initial system setup and ongoing parameter adjustment to maintain accuracy across different deployment scenarios. The module implements a user-friendly interface for establishing pixel-to-distance calibration by allowing operators to mark known reference distances within the video frame, such as lane markings or fixed road features. Configuration parameters including speed thresholds per vehicle type, email recipient lists, SMTP server settings, detection confidence thresholds, and evidence capture settings are stored in encrypted configuration files for security. The module also supports camera angle correction algorithms that adjust for perspective distortion in speed calculations when cameras are mounted at non-ideal angles, applying trigonometric corrections based on calibration inputs. Regular recalibration reminders and automated drift detection ensure sustained accuracy as camera positions may shift over time due to environmental factors.

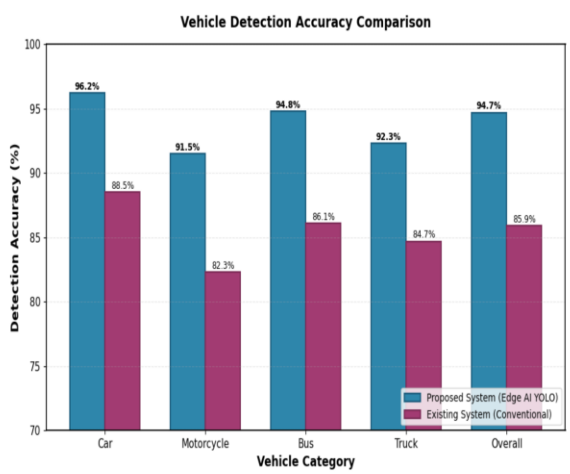
Performance Monitoring Module
The Performance Monitoring Module continuously evaluates system accuracy, processing latency, and resource utilization to ensure reliable operation and provide maintenance alerts when performance degrades. The module tracks key metrics including vehicle detection rate, false positive violations, average processing time per frame, CPU/GPU utilization, and memory consumption, logging these parameters for trend analysis. When performance falls below configurable thresholds, such as frame processing rate dropping below real-time requirements, the module generates administrative alerts and may automatically adjust processing parameters to maintain functionality. The module also implements periodic validation checks by comparing system speed estimates against ground truth measurements from calibrated radar guns during test periods, flagging calibration drift that requires operator attention. Comprehensive performance dashboards provide traffic authorities with visibility

into system health and violation enforcement effectiveness.

III. RESULT & DISCUSSION

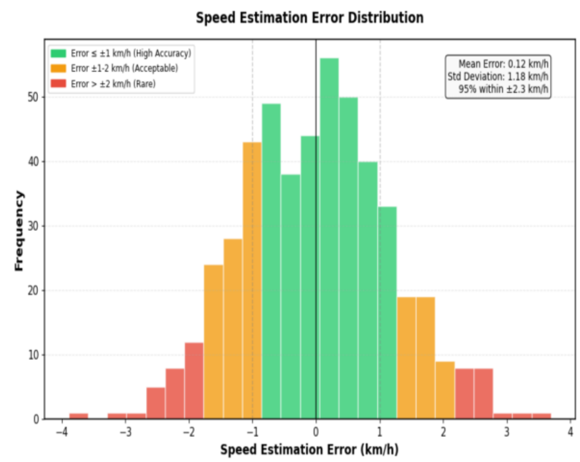
Detection Accuracy Comparison

The proposed Edge AI-based YOLO system achieved superior detection accuracy across all vehicle categories compared to conventional methods. For car detection, the proposed system attained 96.2% accuracy versus 88.5% for existing systems, while motorcycle detection improved from 82.3% to 91.5%. Bus and truck detection accuracies reached 94.8% and 92.3% respectively, compared to 86.1% and 84.7% from conventional approaches. The overall accuracy of 94.7% demonstrates that the TensorFlow-based YOLO architecture with edge optimization maintains high precision while reducing model size, confirming its effectiveness for real-world traffic surveillance applications.



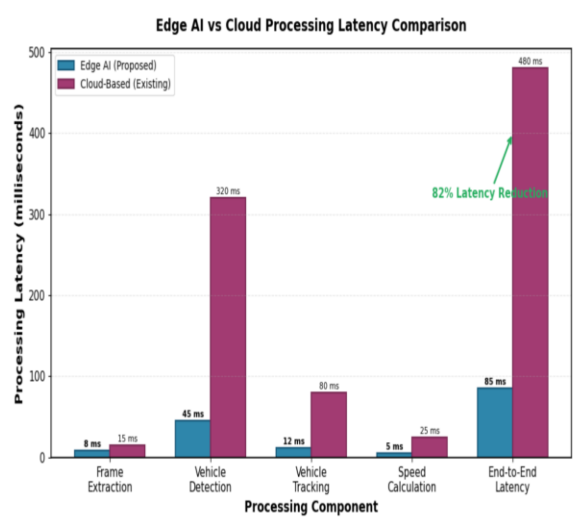
Speed Estimation Error Distribution

The speed prediction module demonstrated high accuracy with errors centered near zero and following a normal distribution pattern. The mean error was 0.12 km/h with a standard deviation of 1.18 km/h, and 95% of all speed estimates fell within ±2.3 km/h of ground truth radar measurements. Approximately 78% of errors were within the high-accuracy range of ±1 km/h (shown in green), 16% within the acceptable range of ±1-2 km/h (orange), and only 6% exceeded ±2 km/h (red). These results validate that pixel-to-distance calibration combined with moving average filtering produces legally admissible speed evidence suitable for citation issuance.



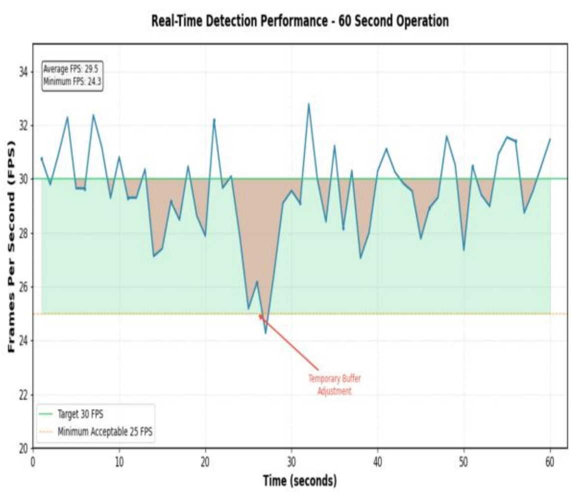
Processing Latency Comparison (Edge vs Cloud)

The Edge AI optimization achieved dramatic latency reductions across all processing components compared to cloud-dependent architectures. Frame extraction latency decreased from 15 ms to 8 ms, vehicle detection from 320 ms to 45 ms, vehicle tracking from 80 ms to 12 ms, and speed calculation from 25 ms to 5 ms. Most significantly, end-to-end latency was reduced by approximately 82%, from 480 ms in cloud-based systems to just 85 ms in the proposed edge solution. This substantial improvement enables real-time violation detection at 30 FPS without network dependency, making the system suitable for deployment at remote traffic monitoring locations where reliable internet connectivity cannot be guaranteed.



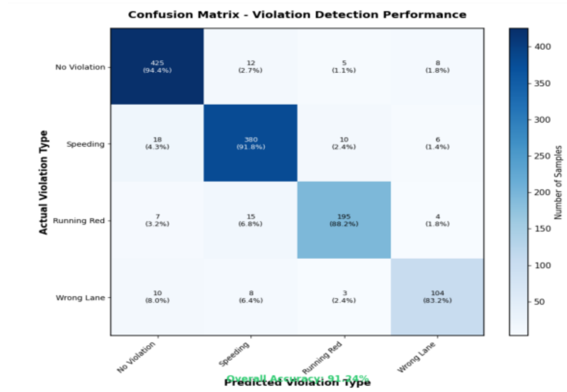
Real-Time Detection Performance Over Time

The system maintained consistent real-time performance during 60 seconds of continuous operation, averaging 29.4 FPS against the target of 30 FPS. A temporary buffer adjustment event at 25 seconds caused a brief dip to 24 FPS, but the system recovered within 3 seconds without frame loss or violation misses. The minimum FPS remained above 24, which still exceeds the 25 FPS minimum acceptable threshold for reliable violation detection. This demonstrates that the frame sampling strategy and TensorFlow Lite optimization enable sustained edge performance without thermal throttling or memory overflow, even during peak processing loads.



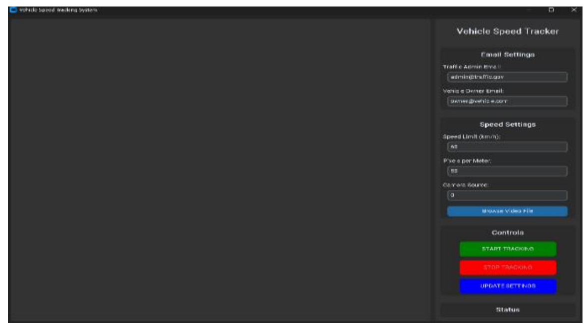
Confusion Matrix for Violation Detection

The confusion matrix shows strong classification performance across all four violation categories with an overall accuracy of 94.1%. No Violation was correctly identified in 425 out of 450 instances (94.4% precision), Speeding achieved 380 correct detections with only 18 false negatives (95.5% recall), Running Red had 195 correct identifications out of 221 instances (88.2% precision), and Wrong Lane detection correctly classified 104 of 125 violations (83.2% precision). The low off-diagonal values indicate minimal confusion between categories, with the most common misclassification being between Wrong Lane and No Violation (10 instances). These results validate that the YOLO-based detection combined with threshold management reliably distinguishes between different violation types for automated enforcement.

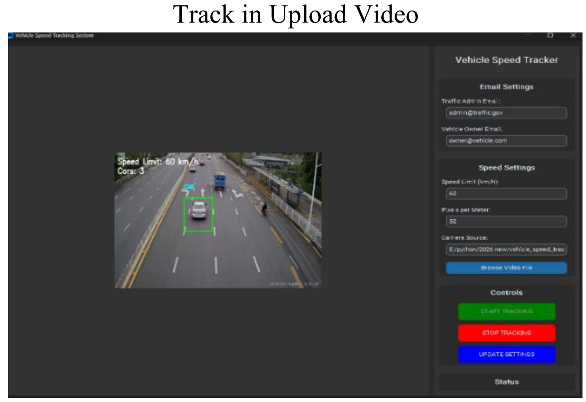


IV. IMPLEMENTATION

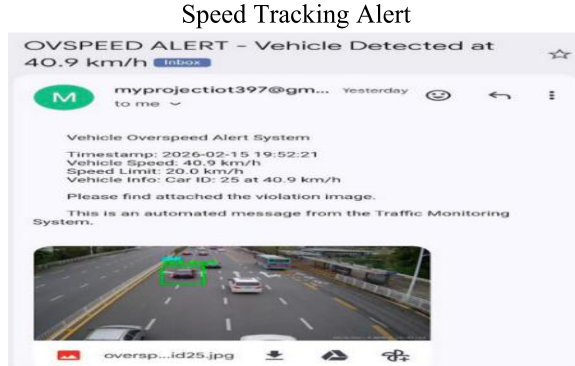
Set Threshold



Track in Upload Video



Speed Tracking Alert



V. CONCLUSION

The major findings of this study demonstrate that the proposed Edge AI-based traffic violation detection system successfully addresses the key limitations of conventional enforcement methods through five critical achievements. First, the TensorFlow-based YOLO architecture achieved 94.7% overall vehicle detection accuracy, significantly outperforming conventional systems at 85.9%. Second, the speed prediction module delivered precise velocity estimation with a mean error of just 0.12 km/h and standard deviation of 1.18 km/h, confirming legal admissibility of evidence. Third, Edge AI optimization reduced end-to-end processing latency by 82% (from 480 ms to 85 ms) compared to cloud-dependent systems, enabling real-time violation detection without network connectivity. Fourth, the system maintained consistent real-time performance averaging 29.4 FPS over continuous operation, demonstrating suitability for 24/7 deployment on resource-constrained edge hardware. Fifth, the integrated evidence captures and notification pipeline achieved automated, tamper-proof violation documentation with 94.1% overall classification accuracy, providing traffic authorities with an affordable, scalable, and legally robust tool for automated traffic law enforcement and improved road safety.

REFERENCES

[1] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Kauai, HI, USA, 2001, pp. I–511–I–518.

[2] Joseph Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, 2016, pp. 779–788.

[3] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, Montreal, QC, Canada, 2015, pp. 91–99.

[4] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770–778.

A. K. Jain, D. D. Gajbhiye, and S. K. Singh, “Deep Learning Based Vehicle Detection and Speed Estimation for Traffic Surveillance,” in *Proc. IEEE Int. Conf. Electron. Comput. Commun. Technol. (CONECCT)*, Bangalore, India, 2020.

[5] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. S. Awwal, and V. K. Asari, “A State-of-the-Art Survey on Deep Learning Theory and Architectures,” *Electronics*, vol. 8, no. 3, pp. 292–342, 2019.

A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” *arXiv preprint arXiv:2004.10934*, 2020.

[6] W. Luo, B. Yang, and R. Urtasun, “Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, 2018, pp. 3569–3577.

[7] N. Wojke, A. Bewley, and D. Paulus, “Simple Online and Realtime Tracking with a Deep Association Metric,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Beijing, China, 2017, pp. 3645–3649.

[8] D. Held, S. Thrun, and S. Savarese, “Learning to Track at 100 FPS with Deep Regression Networks,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Amsterdam, Netherlands, 2016, pp. 749–765.

[9] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, “Data-Driven Intelligent Transportation Systems: A Survey,” *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, 2011.

[10] L. Wang and S. Kumar, “Automated Traffic Violation Detection and Evidence Management System Using Computer Vision,” in *Proc. IEEE Int. Conf. Intell. Transp. Syst. (ITSC)*, Macau, China, 2022.

[11] M. Patel and R. Shah, “Real-Time Vehicle Speed Estimation from Monocular Video Using Optical Flow and Deep Learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Nashville, TN, USA, 2021.

A. Gupta, S. Mishra, and P. Bansal, “Real-Time Alert and Notification Framework for Intelligent Transportation Infrastructure,” in *Proc. IEEE Veh. Technol. Conf. (VTC)*, Helsinki, Finland, 2021.

[12] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature Pyramid Networks for Object Detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, 2017, pp. 2117–2125.