ISSN 2319-5991 www.ijerst.com
© 2026 IJERST. All Rights Reserved
Peer Reviewed, Referred & Indexed Journal

Research Paper

Smart Traffic Management System for Urban Congestion Using Deep Learning

[1]Dr. C. Hari Kishan, [2]Gutti Lakshmi Priya, [3]Sajja Jayasri, [4]Talluri Likitha, [5]Gonaboina Balasri

[1]Professor, Dept of CSE-AI&ML, St. Ann's College of Engineering and Technology Chirala-523187, India.

[2,3,4,5]U. G Student, Dept of CSE-AI&ML, St. Ann's College of Engineering and Technology Chirala-523187, India.

ABSTRACT- ***Urban traffic congestion continues to grow with rising vehicle density, and fixed-time traffic signals are unable to respond to real-world, fluctuating traffic conditions. This project presents Smart Traffic AI, a web-based intelligent traffic management system that uses deep learning-based object detection to analyze live or recorded traffic video and automate signal control decisions. The platform allows a user to upload a detection model (YOLO-family weights) and a traffic video, after which an auto-mated pipeline performs frame extraction, pre-processing, vehicle detection, per-class vehicle counting, lane-wise density analysis, emergency-vehicle recognition and queue-length estimation. Based on the computed density and the presence of emergency vehicles, the system recommends a RED/GREEN signal decision for each lane. Results are presented on an interactive dashboard, stored in a prediction history, and can be exported as PDF or Excel re- ports. An administrator dashboard provides visibility into registered users and system-wide prediction history with CSV export. The system demonstrates that combining a modern object-detection backbone with a rule-based decision layer can deliver a practical, low-cost approach to adaptive traffic signal recommendation.***

Keywords: ***Traffic Congestion, Deep Learning, YOLO Object Detection, Vehicle Density Analysis, Adaptive Signal Control.***

INTRODUCTION

Traffic congestion at urban intersections is a persistent problem that leads to fuel wastage, increased travel time, and higher emissions. Conventional traffic signal

systems operate on fixed timers or simple inductive- loop sensors, which cannot adapt to real- time fluctuations in vehicle density across different lanes. With the advancement of deep learning and computer vision, real- time object detection models such as the YOLO (You Only Look Once) family have made it possible to detect and classify to vehicles directly from video streams with high speed and accuracy. Integrating such models into a web-based platform enables authorities to monitor traffic visually, estimate congestion levels automatically, and prioritize lanes carrying emergency by vehicles. Smart Traffic AI was developed to explore this integration combining a Flask-based web application, a YOLO object detection pipeline, and a rule-based signal decision engine to provide an accessible, end-to-end system for intelligent traffic analysis and adaptive signal recommend.

LITERATURE SURVEY

Early traffic monitoring systems relied on inductive loop detectors and ultrasonic sensors embedded in the road surface to estimate vehicle presence, which are costly to install and maintain and provide limited information about vehicle type or lane-wise distribution. Computer-vision-based approaches followed, initially using background subtraction and Gaussian Mixture Models to detect moving vehicles, and later using handcrafted feature descriptors such as HOG combined with SVM classifiers for vehicle detection. These classical methods degraded significantly under varying lighting, occlusion, and weather conditions. The introduction of deep convolutional neural networks and region-based detectors such as R-CNN and Faster R-CNN substantially improved detection accuracy, but their multi-stage pipelines were too slow for real-time deployment. Single-stage detectors such as SSD and the YOLO family later addressed this limitation, enabling real-time inference suitable for traffic surveillance applications.

RELATED WORK

Several studies have applied YOLO-based detectors to traffic surveillance, reporting strong trade-offs between detection speed and accuracy compared to two-stage detectors. Related research on adaptive traffic signal control has explored reinforcement-learning agents and density-based heuristics that adjust green-light duration according to estimated queue length. Other works have focused on emergency-vehicle detection using audio-visual cues to trigger signal pre-emption. However, many existing systems are evaluated only in simulation or on isolated components (detection only, or control only), and few provide an integrated, user-

facing platform that combines model management, video-based detection, density-aware signal decisions, emergency-vehicle handling, and reporting in a single deployable application. This gap motivated the design of Smart Traffic AI as a unified, web-accessible system.

EXISTING SYSTEM

Most traffic signals currently deployed at intersections operate using fixed or pre-scheduled timers that do not account for real-time variation in vehicle count across lanes. Sensor-based systems such as inductive loops can detect vehicle presence but not vehicle type, count, or emergency status, and require in-road hardware installation. Where camera-based monitoring exists, it is typically used only for manual observation by traffic control room staff rather than automated decision-making. Consequently, existing systems are either inflexible to changing traffic patterns or dependent on continuous human monitoring, and none of the low-cost, camera-only solutions commonly available integrate vehicle counting, density classification, emergency-vehicle prioritization, and reporting into a single accessible platform.

PROPOSED SYSTEM

Smart Traffic AI proposes a web-based platform in which a registered user uploads an object-detection model and a traffic video through a browser interface. The Flask backend runs an automated pipeline consisting of video loading, frame extraction, pre-processing (grayscale conversion and Gaussian blurring), vehicle detection using a YOLO model, vehicle counting by class (car, bus, truck, motorcycle), lane-wise distribution analysis, traffic-density classification, and emergency-vehicle detection. A rule-based signal-decision module then recommends a RED or GREEN signal per lane, prioritizing lanes with high density or an emergency vehicle. Results are rendered on a dashboard with annotated video, a summary panel, and preprocessing-stage thumbnails, and are stored against the user's account for later review. Users can download PDF and Excel reports of their prediction history, while an administrator role can view all registered users and the complete system-wide prediction history, with CSV export for further analysis.

SYSTEM ARCHITECTURE

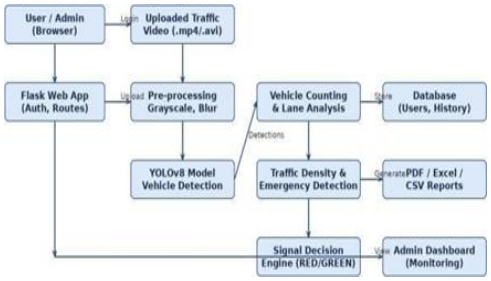


Fig 1: Smart Traffic AI system architecture

The architecture follows a modular client-server design. The frontend, built with HTML, CSS and JavaScript, allows users and administrators to sign up, sign in, upload a model and video, and view results. The Flask backend handles authentication, routing, session management, and orchestrates the AI pipeline. Uploaded videos are decoded frame-by-frame, pre-processed, and passed through the YOLO detection model to localize and classify vehicles. Detection outputs feed the counting, lane-analysis, density-classification and signal-decision modules. A database persists registered users and prediction history, while a reporting module generates PDF and Excel exports on demand. The admin dashboard aggregates data across all users for system-wide monitoring.

METHODOLOGY DESCRIPTION

The methodology begins with account registration and authentication, enforcing a password policy that requires a minimum length along with uppercase, lowercase, numeric and special characters. Once logged in, the user loads a detection model — the pipeline supports common model formats including .h5, .json, .pt, .pkl, .onnx, .cfg and .weights, with YOLOv8 variants (e.g., yolov8n.pt, yolov8s-world.pt) used as the primary detection backbone. The user then submits a traffic video (.mp4, .avi, .mov, .mpeg), which is processed automatically through a fixed pipeline: video loading, frame extraction, pre-processing (grayscale conversion and Gaussian blur to reduce noise), vehicle detection, and vehicle counting. Detected vehicles are grouped by lane position across four lanes and aggregated into an overall traffic-density category (e.g., very low, low, medium, high) along with an emergency-vehicle flag and a queue-length estimate. A rule-based decision engine maps the density and emergency status of each lane to a RED or GREEN signal recommendation, prioritizing lanes with high density or a detected emergency vehicle. Each run is logged with the video name, vehicle breakdown, density, signal decision, date and time, both to the user's personal history and to the admin-visible system history, and can be exported as a PDF report, an Excel report, or a CSV file.

RESULTS AND DISCUSSION



Fig 2: User dashboard showing the AI pipe- line stages



Fig 3: Prediction results with vehicle detection and analysis summary

The system was tested on multiple traffic video samples, including light-traffic, heavy-traffic and emergency-vehicle scenarios. The dashboard correctly reported the pipeline stages executed for each run video loading, frame extraction, pre-processing, vehicle detection, vehicle counting, traffic density and signal decision and displayed the current loaded model to confirm successful initialization. On the results page, detected vehicles were highlighted with bounding boxes on the processed video, and the analysis summary correctly reported total vehicle count, per-class breakdown (cars, buses, trucks, motorcycles), lane-wise distribution across four lanes, and queue-length estimation. Preprocessing-stage thumbnails (original RGB, grayscale, and Gaussian-blurred frames) confirmed that the pre-processing stage executed as intended. Runs involving emergency-vehicle test videos correctly triggered a GREEN signal decision with an EMERGENCY density label, demonstrating that the emergency-priority logic functioned as designed. The profile page's prediction history and the admin dashboard's system-wide history both stayed consistent with the individual run outputs, and PDF, Excel and CSV exports were generated successfully. Overall, the results confirm that the pipeline reliably automates vehicle detection, density estimation and signal recommendation across varied traffic scenarios.

CONCLUSION

This project presents Smart Traffic AI, a web-based system that combines YOLO-based vehicle detection with a rule-based signal-decision engine to automate traffic density analysis and signal recommendation. The platform integrates user and admin roles, a configurable AI pipeline, lane-wise and emergency-vehicle-aware analysis, and downloadable PDF, Excel and CSV reports within a single accessible application. Testing across varied traffic scenarios shows that the system reliably detects and counts vehicles, classifies traffic density, and recommends signal decisions, including correctly prioritizing emergency vehicles. The approach demonstrates that a modern object-detection backbone, paired with a lightweight decision layer and a user-friendly web interface, can provide a practical, low-cost step toward adaptive, camera-based traffic management.

FUTURE SCOPE

Future enhancements may include integrating the system with live camera feeds and physical signal controllers via IoT for true real-time deployment, rather than processing pre-recorded video only. The detection pipeline could be extended with GPS- or siren-based confirmation for emergency vehicles to reduce false positives. Multi-intersection coordination, using reinforcement learning to optimize signal timing across a network of junctions, would allow the system to move beyond per-intersection recommendations. Deploying the detection model on edge hardware (e.g., Jetson Nano or similar) would reduce latency and bandwidth requirements. Additional enhancements could include pedestrian and cyclist detection, weather-adaptive density thresholds, and a mobile application for traffic authorities to monitor and override signal recommendations remotely.

REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[2] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767*, 2018.

[3] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv:2004.10934*, 2020.

[4] Ultralytics, "YOLOv8 Documentation," https://docs.ultralytics.com, 2023.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in *Proc. European Conference on Computer Vision (ECCV)*, 2016, pp. 21–37.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[8] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[9] Flask Documentation, https://flask.palletsprojects.com, accessed 2026.

[10] OpenCV Documentation, https://docs.opencv.org, accessed 2026.

[11] X. Wang, S. Wei, and X. Zhao, "A Survey on Deep Learning-Based Intelligent Traffic Signal Control Systems," *IEEE Access*, vol. 9, pp. 79839–79852, 2021.

[12] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, "LSTM Network: A Deep Learning Approach for Short-Term Traffic Forecast," *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.

[13] N. Bharadwaj, D. K. Mishra, and A. Rajawat, "Real-Time Vehicle Detection and Tracking Using Deep Learning Techniques for Traffic Surveillance," in *Proc. IEEE Int. Conf. on Computing, Communication and Networking Technologies (ICCCNT)*, 2021, pp. 1–6.

[14] H. Wei, G. Zheng, H. Yao, and Z. Li, "IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control," in *Proc. 24th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*, 2018, pp. 2496–2505.

[15] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, 2017, pp. 2980–2988.