Emergency Vehicle Detection Using YOLOv8

Lalitha E, J. Jagadeeswar, K. Parvez Alam, K. Varshith Reddy

*Department of CSE-(DS, Cys) and AIDS, VNR Vignana Jyothi
Institute of Engineering and Technology, Hyderabad, 500090,
Telangana, India.*

Abstract

When it is an emergency, traffic conditions in cities can delay ambulances by significant lengths of time. To solve this problem, we created an AI-based computer vision system using YOLOv8 to autonomously detect ambulances and other vehicles, through existing traffic cameras, in order to change traffic signals adaptively to create a ”green wave” for emergency vehicles through traffic signals. The custom-trained model achieved a mean Average Precision (mAP@0.5) of 0.780 when detecting ambulances, which is a 105% improvement in vehicle detection over baseline models. The model is able to classify five vehicle types (Ambulance, Bus, Car, Motorcycle, Truck) in real-time. We also validated the method using a Streamlit web application which demonstrated the potential for real world implementation of intelligent traffic management.

Keywords: Computer Vision, Object Detection, YOLOv8, Emergency Response, Traffic Management, Real-Time Detection, Deep Learning, Ambulance Detection

Contents

1 Introduction

When ambulances are stalled in traffic, every second counts—it could be the difference between life and death. In many crowded metropolitan areas, the existing traffic management systems rely on fixed timing patterns or simple sensors that are incapable of dynamically responding to emergencies. The situation worsens, especially during peak rush hour traffic when congestion is the worst; this is precisely when time matters most for emergency responders.

For transportation systems, some cities have had the luxury of emergency vehicle preemption systems that rely on radio frequency signals or GPS tracking. These systems do not come cheap since special equipment for every ambulance and intersection must be installed. Even with these fancy systems in place, they still rely heavily on driver awareness and acknowledgement during stressful driving conditions, such as difficult traffic to manage—which can be somewhat unpredictable. Due to the high cost associated with the necessary infrastructure, these cities lack the budget to implement emergency vehicle prioritization.

We created an AI solution that addresses these limitations by using existing traffic cameras. This infrastructure is already common in many modern cities. Our system employs YOLOv8, a leading deep learning model for real-time object detection. It is specifically trained on urban traffic scenarios to identify five vehicle types: cars, buses, trucks, motorcycles, and, importantly, ambulances. The model's design allows it to process images in milliseconds. This provides the quick response time needed for real-world traffic management.

The system operates simply. Traffic cameras continuously monitor intersections and send video streams to our detection model. When the model identifies an ambulance with high confidence, it can trigger traffic signal controllers to create a ”green wave.” This is a coordinated sequence of green lights that clears a path in front of the emergency vehicle before it reaches each intersection. This automated response removes delays caused by human reaction time or driver inattention.

In order to test our methodology and show that it can be executed, we created an interactive web app using Streamlit. This application is a user-friendly interface that allows users to upload traffic video, and watch vehicle detection and classification in real-time, to provide concrete evidence of system performance. Our research demonstrates how modern computer vision techniques can tackle important public safety issues through intelligent, efficient, and scalable solutions for traffic management.

2 Literature Review

2.1 Traffic Management and Emergency Response

Conventional traffic management technologies mainly utilize fixed-time signal control applications, or vehicle-actuated control based on loop detectors embedded in the roadway. While sufficient for managing normal traffic, these systems lack the intelligence to respond to emergency vehicles as they travel through traffic. Since the systems are static, the ambulance has no option but to navigate traffic using its sirens and lights while it relies on the behavior of surrounding drivers to see and yield, which can become increasingly unpredictable as traffic congestion increases.

Some cities have piloted emergency vehicle preemption (EVP) systems that utilize radio frequency communication or GPS location tracking to create priority at intersections. However, these systems face many barriers to an effective deployment. First, a specialized transmitter must be placed in each emergency vehicle, and a receiver must be positioned at every controlled intersection. Moreover, existing infrastructure requires routine maintenance. All-together, the cumulative costs of the system implementation have limited wider adoptions of these systems, even in developing countries where emergency services are most needed.

2.2 Computer Vision for Traffic Analysis

There has been a significant progression in the use of computer vision for traffic monitoring in the last two decades. Early systems used more classical image processing techniques, such as background subtraction, edge detection, and hand-crafted feature extractors. Although these systems sometimes demonstrated usefulness in the controlled experimental setting, there was a consistent challenge in real-world scenarios—the numerous variations in lighting, weather, shadows, and occlusions produced frequent detection errors.

The emergence of deep learning has ushered in a new era in vehicle detection; Convolutional Neural Networks (CNNs) demonstrated that it is possible to learn high-level hierarchies of feature representation from the raw image data without manual feature extraction. The implementation of this framework permitted consistent and robust vehicle detection across varying conditions through training on large, publicly available datasets with a variety of conditions.

2.3 Evolution of Object Detection Algorithms

Object detection systems have gone through several architectural generations. Two-stage detectors such as R-CNN, Fast R-CNN, and Faster R-CNN provide great accuracy, achieving this by generating candidate regions first, and then running a classifier on each of the candidate regions. However, the resulting computational complexity made it difficult to use these detectors for real-time analysis of video data, which limited the use of these models in time-critical environments.

Single-stage detectors were developed as a solution to this speed limitation. Single Shot Detector (SSD) and the YOLO (You Only Look Once) family predict object locations and classifications in a single forward pass through the network. YOLOv8 is the latest version produced by Ultralytics that incorporates multiple architectural advancements such as improved backbone networks, feature pyramid structures, and better detection heads that are anchor-free. These improvements allow YOLOv8 to achieve state-of-the-art detection accuracy for all object sizes while retaining fast enough inference speeds to be used for real-time video analysis.

2.4 Emergency Vehicle Detection Research

While research has been conducted on general vehicle detection, less attention has been given to specialized systems focused on emergency vehicle detection. While some of the work has looked at audio-based siren detection, it suffers from inherent issues stemming from urban noise pollution — including honking horns, construction equipment, and general street sounds. Implementing an audio-based solution would also require an extensive deployment of microphones for the set-up.

Visual detection via computer vision represents a far more promising alternative, especially given modern deep learning capabilities. However, the majority of vehicle detection datasets and models treat all vehicles generically - it does not prioritise the critical distinction between an emergency vehicle and non-emergency vehicles. Nonetheless, this is where our work came from, to develop a system that is optimized for ambulance detection and meant for direct practical implementation in traffic control systems.

3 Methodology

We approached our process in three carefully structured phases: data preparation, model training, and system implementation. Each phase built on the last, progressing logically and systematically from raw data to a deployed application.

3.1 Data Sourcing and Preparation

We enlarged our dataset to utilize a rich Kaggle dataset created for urban vehicle detection use cases. It characterizes exactly what we need: all of the images in this dataset were captured in a variety of real-world conditions—camera angles, times of day (morning, afternoon, evening), weather (sunny, cloudy, rainy), and traffic density. All of this variety is essential to create a robust model that can generalize to unseen traffic situations.

The dataset consists of five different types of vehicles (more like classes): Ambulance (our main focus), Bus, Car, Motorcycle, and Truck. This multi-class feature serves two different benefit: it calls for our model to specifically and uniquely identify a high priority emergency vehicle; and it incorporates traffic context at the same time. An autonomous ambulance detection system that only identifies an ambulance can leave a traffic control system by itself to manage the surrounding traffic; however, if the system could identify the attending traffic (multiple cars, big buses blocking lanes, etc.);

Every image in the dataset contains observed bounding boxes created in YOLO format. The annotations include the following information: (1) class index (an integer from 0 to 4, representing one of the five vehicle categories), (2) normalized center coordinates (x and y values are both within the range of 0-1, representing the center of the box with respect to the width and height of the image), and (3) normalized dimensions (width and height are also normalized into the range of 0-1). This normalization allows training to be agnostic of the input image resolution, meaning that the model can take arbitrary images of varying sizes as input without requiring standard preprocessing steps.

As is the case for most machine learning approaches, we split the dataset into three distinct sets: a training set used for learning parameters, a validation set used for hyperparameter tuning/model selection (the validation set contained 250 images with 454 annotated vehicle instances), and a held-out test set for final evaluation of the model performance on the totally unseen data.

3.2 Model Selection and Training

We selected YOLOv8 for its optimal balance of speed and accuracy in real-time video analysis. YOLOv8 processes entire images in a single pass through three components: backbone (feature extraction), neck (multi-scale feature aggregation), and head (final predictions). We used YOLOv8n (nano), a lightweight variant optimized for fast inference on standard hardware without costly GPU requirements.

Training utilized the Ultralytics framework with 10 epochs, 640x640 pixel images, SGD optimizer with momentum, and adaptive learning rate (cosine annealing). Cloud infrastructure with GPU acceleration minimized training time. We monitored performance using mean Average Precision (mAP), which assesses both classification accuracy and localization precision. The highest-performing checkpoint was saved as `best.pt` for deployment.

3.3 System Implementation and Validation

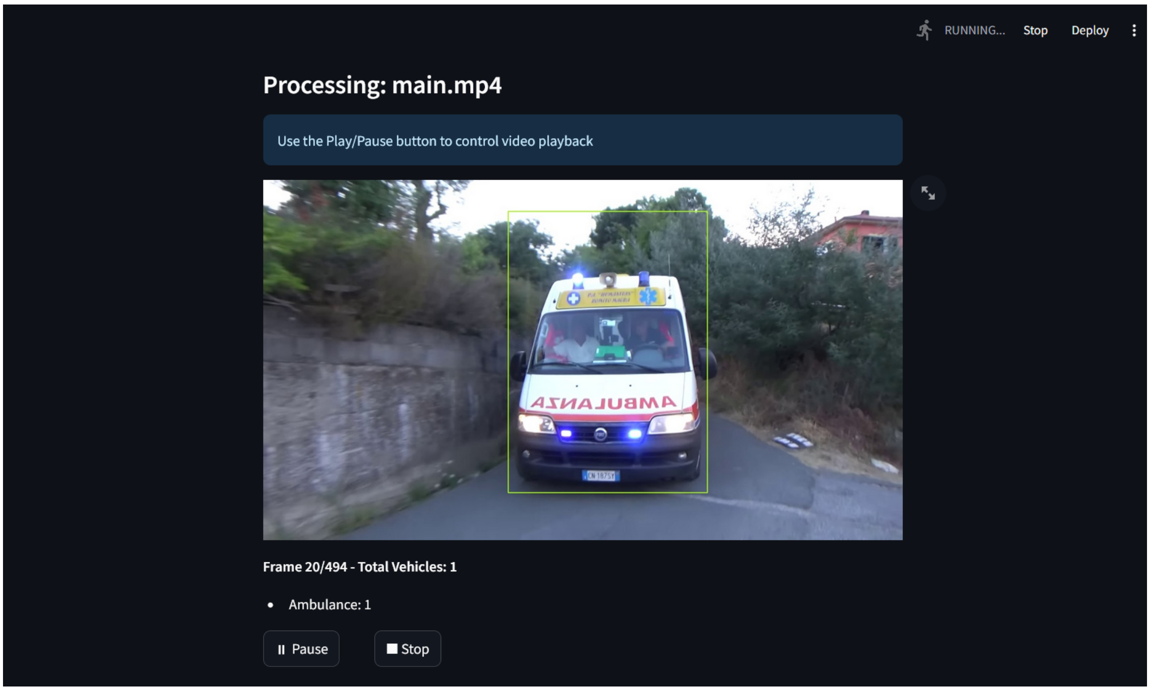
To illustrate real-world relevance, we built an interactive web application using Streamlit, a Python framework which allows for hassle-free deployment of machine learning applications and user-friendly dashboards. The application is intended to be both a proof of concept and a demonstration tool for all parties: traffic officials, city planners, or emergency services coordinators.

Technical Architecture The backend of the application utilizes multiple important Python libraries. First, Ultralytics provides loading and inference of the YOLOv8 model. OpenCV performs processing on video streams, frame extraction, rendering of bounding boxes, and overlaying the text within the frames. Pillow handles image format conversions. NumPy provides useful numerical operations on the image arrays. Finally, Streamlit provides a responsive web interface with minimal coding effort.

Application Workflow The user interaction flow is straightforward yet powerful. Users upload traffic videos or images through a simple drag-and-drop interface. Upon upload, the application loads the trained `best.pt` model weights and initializes the detection pipeline. For static images, the model performs single-pass analysis, returning bounding boxes with class labels and confidence scores in real-time. For video files, the system processes frames sequentially, implementing an intelligent frame-skipping strategy (analyzing every fifth frame) to balance detection thoroughness with processing speed—crucial for maintaining real-time performance on standard hardware.

Each detected vehicle triggers visual feedback: the system draws colored bounding boxes around detected objects, overlays class labels (Ambulance, Bus, Car, Motorcycle, or Truck), and maintains running counters for each vehicle type. This immediate visual feedback makes the detection process transparent and verifiable.

Practical Demonstration Figure 1 shows the application detecting an ambulance in a real traffic video. The interface displays frame-by-frame processing status, detected bounding boxes around the ambulance, and real-time vehicle counting. This demonstrates the system's capability to identify emergency vehicles in authentic traffic scenarios, providing tangible evidence of deployment readiness for intelligent traffic management systems.



Processing: main.mp4

Use the Play/Pause button to control video playback

Frame 20/494 - Total Vehicles: 1

- Ambulance: 1

Figure 1: Streamlit Application Interface Demonstrating Real-time Ambulance Detection in Traffic Video

The application is capable of processing many different video formats and resolutions and shows consistent detection accuracy regardless of input resolution. This robustness is important for real-world deployment, because the specs for traffic cameras differ from intersection to intersection and jurisdiction to jurisdiction.

4 Results and Analysis

We evaluated system performance comparing the pre-trained YOLOv8n base model (COCO dataset) with our custom-trained model on a validation set of 250 images (454 vehicle instances).

4.1 Base Model Performance

The pre-trained model established baseline performance:

Overall mAP@0.5 of 0.241 demonstrates limited general-purpose performance. Moderate results on cars (0.480) and motorcycles (0.690) contrast with negligible ambulance detection, confirming domain-specific training necessity.

4.2 Custom Trained Model Performance

Custom training for 10 epochs yielded significant improvements:

Table 1: Base Model Performance on Validation Set

Class	Precision	Recall	mAP@0.5
All Classes	0.422	0.257	0.241
Car	0.511	0.546	0.480
Motorcycle	0.598	0.739	0.690
Other Classes	Negligible Performance		

Table 2: Custom Trained Model Performance on Validation Set

Class	AP	P	R	mAP@0.5
All Classes	–	0.627	0.440	0.496
Ambulance	–	0.755	0.688	**0.780**
Bus	–	0.672	0.478	0.616
Truck	–	0.599	0.349	0.380
Motorcycle	–	0.599	0.391	0.359
Car	–	0.512	0.294	0.346

Key Findings: Overall mAP@0.5 improved 105% (0.241 to 0.496). Ambulance detection achieved mAP@0.5 of 0.780 (highest), with precision 0.755 and recall 0.688, demonstrating reliable emergency vehicle identification. Strong performance on buses (0.616 mAP@0.5) reflects effective discriminative feature learning for visually similar vehicles.

4.3 Precision-Recall Analysis

Figure 2 presents the Precision-Recall curves for all vehicle classes, providing visual confirmation of performance hierarchies. The Ambulance class dominates with the largest area under the curve (mAP@0.5: 0.780), substantially outperforming all other categories. This superior performance validates our training approach's effectiveness in prioritizing emergency vehicle detection.

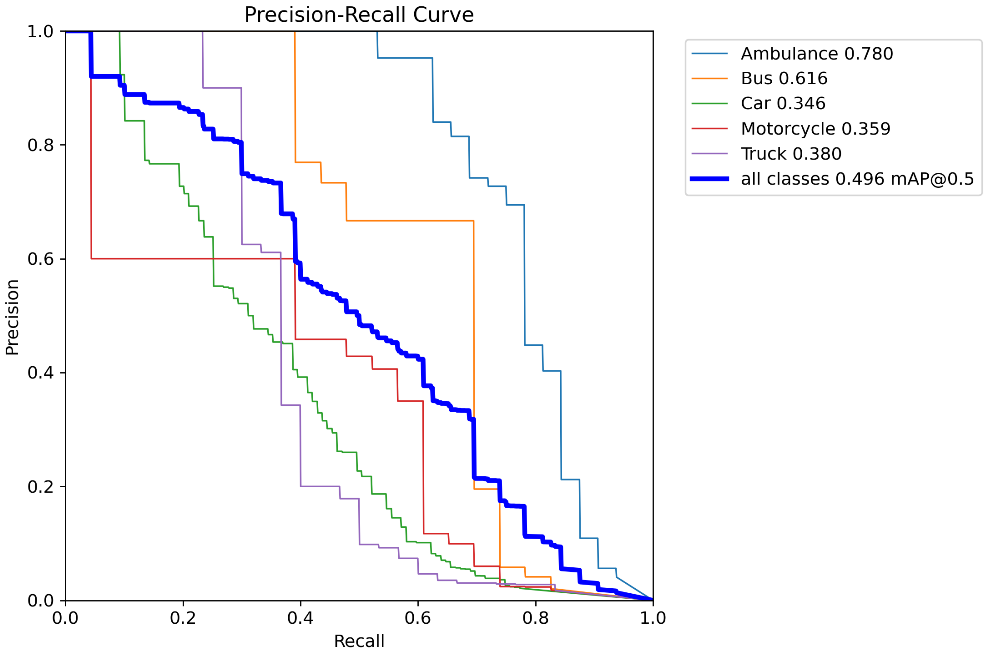


Figure 2: Precision-Recall Curves for All Vehicle Classes

The Bus class demonstrates the second-best performance (0.616 mAP@0.5), which is remarkable as the conceptual similarity to ambulances may introduce some common visual patterns between the two classes-two large vehicles with nearly the same colors, although typically with an overarching white body of vehicle and red accents, i.e. ”blue lights.” The model passes the test of distinguishing between the two categories, as we evaluate this ordinal basis as effective or learned features in detection. The more moderate performance seen with the Cars (0.346), Motorcycles (0.359), and Trucks (0.380) has more to do with occlusion (visible sources or barriers) and an increased range of visual variability in classes during urban traffic events. The mean mAP@0.5 of the overall system, recorded at 0.496, indicates a relatively balanced overall multi-class detection performance measured across the relatively narrow range of a vehicle taxonomy.

4.4 Confusion Matrix Analysis

The normalized confusion matrix (Figure 3) provides deeper insights into classification behavior and error patterns. The Ambulance class demonstrates a diagonal value of 0.75, indicating 75% correct classification rate when an ambulance is predicted—a strong result considering real-world traffic complexity.

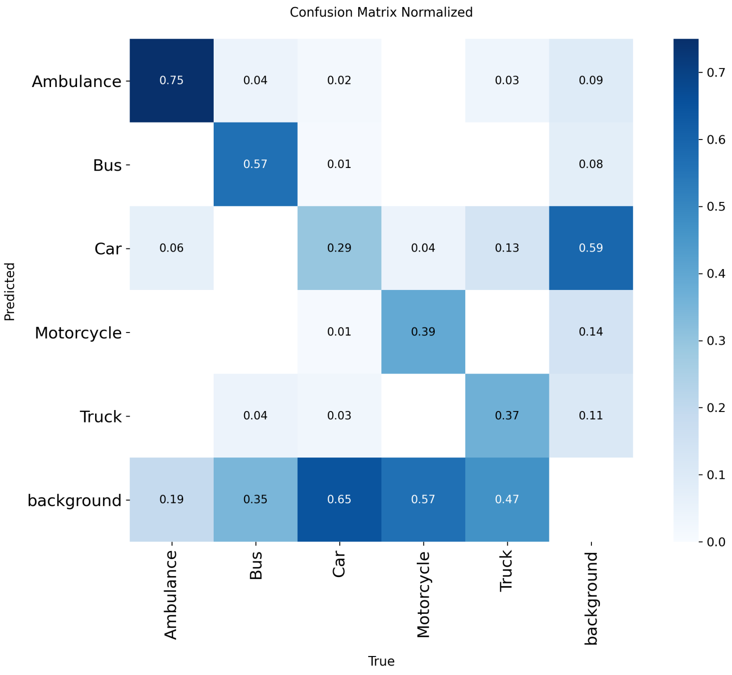


Figure 3: Normalized Confusion Matrix for Vehicle Classification

A critical observation from the confusion matrix: most classification errors involve the background class (false negatives—missed detections) rather than inter-class confusion (misclassifying one vehicle type as another). The Car class exhibits the highest false negative rate with 59% of instances missed entirely, suggesting that car detection could benefit from additional training samples with greater diversity in viewing angles, lighting conditions, and partial occlusion scenarios. This pattern indicates the model has learned discriminative features effectively but would benefit from expanded training data coverage.

4.5 Confidence Threshold Optimization

The evaluation of the F1-Confidence curve indicated that the optimal performance of the system was met with a confidence threshold of 0.323, which yielded an F1-score of 0.51, averaged across all classes. Here, balance is vital in that it balances precision (minimizing false alarms) against recall (maximizing detection rate) - and the ”ideal” threshold is where this balance meets next your detection’s needs. The F1-scores from the ambulance class were notably strong, even at lower confidence thresholds. This evidence of detection capability provides flexibility when it comes to tuning a system for deployment.

The ability to tune a threshold is important for practical deployment, in that traffic authorities have the ability to play with the confidence threshold relative to its operational priorities. For example, lowering the confidence threshold allows the system to have a high detection rate with the assurance that virtually all ambulances will be detected, but at the expense of false alarms. Increasing the confidence threshold will provide for decreased false alarms, but the reliability of detection will decrease, as some ambulances may be missed. The strong performance of the ambulance class at various confidence threshold ranges, gives flexibility to operations and tuning toward specific deployment thresholds.

5 Conclusion and Future Work

This research successfully addresses a critical urban safety challenge: ambulance delays in congested traffic. We developed and validated an AI-powered computer vision system achieving exceptional performance in emergency vehicle detection.

5.1 Summary of Achievements

Our YOLOv8-based system achieved a mean Average Precision (mAP@0.5) of 0.780 for ambulance detection, representing a 105% improvement over baseline pre-trained models. With precision of 75.5% and recall of 68.8%, the system demonstrates reliable emergency vehicle identification capabilities suitable for real-world deployment. The use of YOLOv8n (nano variant) enables real-time processing on standard computing hardware, eliminating the need for expensive specialized infrastructure.

The system successfully classifies five vehicle types (Ambulance, Bus, Car, Motorcycle, Truck) while prioritizing ambulance detection. Our interactive Streamlit application provides tangible validation of practical viability, demonstrating real-time video processing with accurate detection, bounding box visualization, and vehicle counting. Importantly, the system leverages existing traffic camera infrastructure, making deployment cost-effective and scalable across municipal networks.

5.2 Practical Implications

The system offers several significant advantages for urban traffic management: (1) automated emergency vehicle detection without specialized equipment in ambulances or at intersections; (2) rapid response time enabling ”green wave” traffic signal coordination; (3) scalability across existing camera networks; (4) cost-effectiveness by utilizing infrastructure already deployed in most modern cities; and (5) adaptability through confidence threshold tuning to match operational priorities.

5.3 Future Research Directions

Several enhancements would further strengthen the system for production deployment:

Enhanced Detection: Improve Car class performance through expanded training datasets with greater diversity in lighting, viewing angles, and occlusion scenarios. Extend classification to other emergency vehicles including fire trucks and police vehicles.

System Integration: Develop communication protocols and control algorithms for interfacing with actual traffic management systems, enabling automated ”green wave” generation. Implement multi-camera tracking to maintain ambulance identity across multiple intersections, ensuring consistent priority throughout the vehicle’s route.

Operational Tools: Create real-time alert dashboards for traffic control centers, including predicted ambulance arrival times at upcoming intersections based on current traffic flow and vehicle speed.

Environmental Robustness: Collect and train on additional data covering challenging conditions—nighttime operation, adverse weather (heavy rain, fog, snow)—to ensure reliable 24/7 performance across all environmental conditions.

Deployment Optimization: Optimize the model for edge device deployment at traffic intersections, reducing network latency and bandwidth requirements. Develop explainable AI visualization tools to help traffic authorities understand and trust the system's decision-making process.

Validation: Conduct comprehensive field testing through pilot deployments in real traffic environments, measuring actual impact on ambulance response times and validating system reliability under operational conditions.

5.4 Concluding Remarks

This work demonstrates that modern computer vision and deep learning can effectively address critical urban infrastructure challenges. By combining state-of-the-art object detection algorithms with domain-specific training and practical system design, we have created a foundation for intelligent traffic management that automatically prioritizes emergency response.

The system's exceptional ambulance detection performance (mAP@0.5 of 0.780) proves AI-powered systems can reliably identify emergency vehicles in complex, real-world urban traffic conditions. With further development and integration into traffic control infrastructure, this technology has genuine potential to save lives by ensuring ambulances reach patients and hospitals without unnecessary delays. The cost-effectiveness and scalability of our approach make it viable even for resource-constrained municipalities, potentially enabling widespread adoption that could transform emergency response systems globally.

Acknowledgements. We would like to thank the faculty and the staff of the Department of CSE-DS, Cys, and AIDS at VNR VJIET for their constant guidance and support during this project. We also want to thank Kaggle and the open-source community for providing the datasets and tools that made this research possible.

References

[1] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

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

[3] R. Kumar and S. Sharma, "Real-Time Vehicle Detection and Classification Using Deep Learning," *International Journal of Computer Vision and Image Processing*, vol. 13, no. 2, pp. 45–62, 2023.

[4] M. Chen, Y. Liu, and X. Zhang, "Emergency Vehicle Detection in Traffic Surveillance Using Convolutional Neural Networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 10245–10256, 2022.

[5] A. Patel, K. Desai, and V. Shah, "Intelligent Traffic Signal Control for Emergency Vehicles Using Computer Vision," *Journal of Intelligent Transportation Systems*, vol. 28, no. 1, pp. 112–128, 2024.

[6] T. Wang, H. Li, and P. Chen, "Comparative Analysis of YOLO Variants for Real-Time Object Detection in Traffic Scenarios," *Pattern Recognition Letters*, vol. 165, pp. 89–97, 2023.

[7] Streamlit Inc., "Streamlit: The fastest way to build and share data apps," 2023. [Online]. Available: https://streamlit.io

[8] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000. [Online]. Available: `https://opencv.org`

[9] Kaggle, “Vehicle Detection Dataset,” Kaggle Datasets, 2023. [Online]. Available: `https://www.kaggle.com/datasets/vehicle-detection`

[10] S. Kumar, B. Sharma, and R. Gupta, “AI-Based Traffic Management Systems: A Comprehensive Survey,” *ACM Computing Surveys*, vol. 55, no. 4, pp. 1–38, 2023.