AI-Powered Ambulance Detection and Traffic Management System Using YOLOv8

Lalitha E[*] Jagadeeswar Jonnadula[†] K Parvez Alam K Varshith Reddy

Abstract

In medical emergencies, response time is crucial and can be the difference between life and death. However, ambulances often face delays due to traffic in urban areas, which can prevent them from reaching patients or hospitals on time. Standard traffic management systems cannot automatically detect emergency vehicles or adjust traffic signals to help them pass quickly.

In this study, we designed and implemented an AI-powered computer vision system that detects ambulances and other vehicles in real-time using existing traffic cameras. The system uses YOLOv8 (You Only Look Once, version 8), an advanced object detection model trained on a custom dataset of urban traffic scenarios. Our approach creates a "green wave" by communicating with traffic lights to clear paths for ambulances before they reach intersections.

The trained model showed impressive results, achieving a mean Average Precision (mAP@0.5) of 0.780 for ambulance detection, which represents a 105% improvement over the baseline pre-trained model. We developed an interactive web application with Streamlit to demonstrate real-time vehicle detection and counting on images and videos. The system effectively classifies five types of vehicles: Ambulance, Bus, Car, Motorcycle, and Truck, laying a solid foundation for traffic management that prioritizes emergency response.

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

Contents

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

[†]Corresponding author. E-mail: `jagadeeswar079@gmail.com`

1 Introduction

In a medical emergency, every second counts. What if an ambulance is stuck in traffic while transporting a loved one? A few extra minutes could mean the difference between life and death. Our project arose from this urgent issue: how can we create a system that automatically clears a path for ambulances through the city?

Standard traffic management systems are static and do not adapt to emergency situations. While manual traffic control and siren-based priority systems exist, they depend on human intervention and driver awareness, often causing delays that can be deadly. In busy urban areas, these limitations become more evident.

To tackle this pressing issue, we created a computer vision solution powered by AI that uses existing traffic cameras to observe traffic. This system utilizes deep learning to identify emergency vehicles in real-time and can adjust traffic signals to create a clear passage.

The heart of our system uses YOLOv8, a leading object detection model known for its speed and accuracy. We trained this model on thousands of images of busy streets, teaching it to recognize cars, buses, trucks, motorcycles, and, crucially, ambulances. After extensive training, the model quickly identifies emergency vehicles with high accuracy.

To showcase the system's capabilities, we built an interactive web application where users can upload traffic videos and see the AI analyze them in real-time. The system automatically detects each vehicle, draws bounding boxes around them, labels them by type, and keeps a count. The idea is simple: when the system detects an ambulance, it communicates with traffic lights to create a ”green wave,” clearing the path before the ambulance reaches an intersection.

This project illustrates how AI and computer vision can address real-world challenges that impact public safety and emergency response.

2 Literature Review

2.1 Traditional Traffic Management Systems

Standard traffic management systems mainly rely on fixed-time signal control or vehicle-actuated control using loop detectors embedded in the road. While these systems can manage regular traffic, they cannot respond dynamically to emergency vehicles. Some cities have tried emergency vehicle preemption systems using radio signals or GPS, but these require special gear in both vehicles and infrastructure, making them costly to implement widely.

2.2 Computer Vision in Traffic Monitoring

Recent developments in computer vision allow for automated traffic analysis with existing cameras. Early methods used traditional image processing techniques like background subtraction and feature extraction. However, these techniques struggled with changes in lighting, obstructions, and complex traffic conditions. The rise of deep learning has transformed this area, enabling reliable detection and classification of vehicles in various environments.

2.3 Object Detection Algorithms

Modern object detection has evolved with many generations of algorithms. Two-stage detectors like R-CNN and Faster R-CNN achieve high accuracy but are slower. Single-stage detectors like SSD and YOLO emphasize speed while maintaining good accuracy, making them ideal for real-time applications.

The YOLO (You Only Look Once) family of models has become very popular for real-time object detection. YOLOv8, released by Ultralytics, is the latest version with improved structure, better feature extraction, and stronger performance across different object sizes. Its capacity to process images in a single forward pass makes it perfect for video stream analysis needed in traffic monitoring.

2.4 Emergency Vehicle Detection Systems

While there has been considerable research on general vehicle detection, specialized systems for emergency vehicles are still limited. Some methods have attempted audio detection of sirens but face interference from urban noise. Visual detection using deep learning provides more reliable identification, especially when combined with vehicle type classification. However, most existing datasets and models focus on general traffic analysis, not specifically on emergency vehicles.

Our project builds on these foundations by creating a specialized system that detects vehicles with high accuracy and optimizes for ambulance recognition, allowing practical integration with intelligent traffic control systems.

3 Methodology

We organized the development of our AI-powered ambulance detection system into three main phases: (1) Data Sourcing and Preparation, (2) Model Selection and Training, and (3) System Implementation and Validation. This comprehensive approach ensures smooth progression from raw data to a fully functional application ready for real-world deployment.

3.1 Data Sourcing and Preparation

The success of any deep learning model relies on the quality and diversity of its training data. For this project, we used a public dataset from Kaggle tailored for vehicle detection in urban settings. This dataset provided a wide range of images showing various traffic situations, including different times of day, weather types, and camera angles.

The dataset was categorized into five vehicle classes:

- **Ambulance** (priority target class)
- **Bus**
- **Car**
- **Motorcycle**
- **Truck**

This classification was essential for helping the model recognize the high-priority ambulance class and contextualize it within regular traffic.

Each image in the dataset was paired with an annotation file in YOLO format, which indicates the class and location of each object using normalized coordinates. Each annotation line contains:

- Class index (0-4 for the five vehicle types)
- Center x-coordinate (normalized)

- Center y-coordinate (normalized)
- Width (normalized)
- Height (normalized)

The normalized coordinate format makes the training process independent of input image resolution, ensuring consistency across images of varying sizes. The dataset was pre-divided into three subsets:

- **Training set:** Used for training the model parameters
- **Validation set:** Used for tuning hyperparameters and selecting the model (250 images, 454 annotated instances)
- **Test set:** Reserved for final evaluation with completely unseen data

3.2 Model Selection and Training

YOLOv8 Architecture We chose YOLOv8 (You Only Look Once, version 8) as the main detection model for this system. YOLOv8 provides an excellent mix of speed and accuracy, making it suitable for real-time video analysis. As a single-stage detector, it processes the entire image in one go, allowing for significantly faster inference compared to two-stage detectors while keeping high accuracy.

The YOLOv8 architecture includes three main components:

1. **Backbone:** The feature extraction network that processes the input image
2. **Neck:** The feature aggregation module that combines features at different scales
3. **Head:** The detection head that produces final predictions for bounding boxes and class probabilities

We specifically selected the YOLOv8n (nano) variant, a lightweight version optimized for fast inference while maintaining strong detection performance. This choice balances efficiency with accuracy, making the system usable on standard hardware without needing costly GPU resources.

Training Configuration We used the Ultralytics framework for training, which offers an optimized implementation of YOLOv8 with advanced features. Key training parameters included:

- **Epochs:** 10 (full passes through the training dataset)
- **Image size:** 640×640 pixels (standard input resolution)
- **Batch size:** Optimized based on available GPU memory
- **Optimizer:** SGD with momentum
- **Learning rate:** Adaptive with cosine annealing

Training was done on cloud infrastructure with GPU support to reduce training time. All input images were resized to 640×640 pixels, ensuring uniform input sizes that enhance model stability and performance.

Model Evaluation Throughout training, we continuously monitored model performance on the validation set using mean Average Precision (mAP) as the main metric. mAP provides a thorough assessment of detection quality by measuring both accuracy and localization precision.

The model checkpoint with the highest mAP on the validation set was saved as `best.pt`, which represents the best-trained model with optimal generalization capability. We then used this model for testing and deployment.

3.3 System Implementation and Validation

To showcase the trained model's abilities, we created an interactive web application using Streamlit. We chose Streamlit for its quick ability to turn Python scripts into user-friendly web apps, making it great for creating a proof-of-concept demonstration.

Technology Stack The application backend is built using Python with these key libraries:

- **Ultralytics:** Loads the trained `best.pt` model and performs inference
- **OpenCV:** Manages video processing, frame extraction, bounding box rendering, and text overlays
- **Pillow:** Handles image format conversions
- **NumPy:** Provides efficient numerical operations on image data
- **Streamlit:** Builds the interactive web interface

Application Workflow The application follows a simple process:

1. Users upload an image or video file through the web interface
2. The application loads the media file and the trained YOLO model
3. For images:
    - The model analyzes the image in one pass
    - It returns detected objects with class labels, confidence scores, and bounding box coordinates
4. For videos:
    - The application processes the video frame by frame
    - To maintain real-time performance, a frame-skipping technique processes every fifth frame
    - Each processed frame uses the same detection pipeline as images
5. For each detection:
    - A colored bounding box is drawn around the detected vehicle
    - The class label is displayed
    - Vehicle counts are totaled by type
6. Results are shown in the web interface with visualizations and statistics

This implementation provides an intuitive demonstration of the system's capabilities and serves as a starting point for integrating with actual traffic management infrastructure.

4 Results and Analysis

To evaluate our ambulance detection system's performance, we conducted a thorough assessment comparing two models: the pre-trained YOLOv8n base model (trained on the general COCO dataset) and our custom-trained model (fine-tuned on our vehicle-specific dataset). The evaluation used a validation set of 250 images containing 454 annotated vehicle instances.

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		

4.1 Base Model Performance

First, we assessed the pre-trained YOLOv8n model on our validation set to establish a baseline. The base model, trained on the general COCO dataset, was not optimized for our specific vehicle classes.

The base model achieved an overall mAP@0.5 of only 0.241. While it performed moderately on generic cars and motorcycles (mAP of 0.480 and 0.690 respectively), it struggled to detect other vehicle types, including ambulances, with any meaningful accuracy. The Precision-Recall curves for the base model remained close to the origin, indicating poor performance on our specialized dataset.

This result met our expectations and highlighted that a general-purpose object detector, without training specific to our domain, is inadequate for this unique traffic management application.

4.2 Custom Trained Model Performance

After fine-tuning YOLOv8n on our custom dataset for 10 epochs, we saw dramatic improvements in all metrics.

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: **Improved Overall Performance:** The overall mAP@0.5 is 0.496 compared to the base model, reflecting an increase of 105% from 0.241 to 0.496.

Ambulance detection was the prime focus of this project, and thus, the model performed outstandingly for the same:

- **mAP@0.5: 0.780** (highest among all classes)
- **Precision: 0.755** (when the model predicts ”ambulance”, it is correct 75.5% of the time)
- **Recall: 0.688** - the model detects 68.8% of all ambulances present

These metrics indeed show that the model is very reliable in detecting ambulances and is able to find most of the emergency vehicles moving on roads. Also, visually, one can see that the Ambulance class has the biggest area in the Precision-Recall curve.

Strong General Vehicle Detection: It also showed good performance for other types of vehicles, especially buses, with 0.616 mAP@0.5, since their visual features have some similarities with ambulances. It thus shows discriminative features that were learned by the model.

4.3 Confusion Matrix Analysis

The normalized confusion matrix gave further insights into model behavior:

- The Ambulance class had a diagonal value of 0.75, meaning 75% correct classification if an ambulance is predicted
- The class Car had the highest number of false negatives, with 154 instances classified as background (missed detections)
- This explains the relatively low recall and mAP for the Car class and indicates an avenue for future improvement with more diversity in the training data

4.4 Confidence Threshold Analysis

The F1-Confidence curve indicates that the confidence threshold value of 0.323 results in an optimal model balance between precision and recall at an averaged F1-score across all classes of 0.51. For the Ambulance class, the model maintains strong F1-scores even at lower confidence thresholds, indicating robust detection.

This analysis is important for practical deployment, as it enables system administrators to tune the confidence threshold according to operational needs—higher detection rate (lower threshold) or fewer false alarms (higher threshold).

4.5 Demonstration of the Practical System

The Streamlit web application performed real-time detection successfully:

- Processed uploaded images and videos with low latency
- Accurately detected and labeled vehicles using bounding boxes
- Provided live counts of vehicles by type
- Smoothened performance by frame-skipping optimization - processing every 5th frame

It is a proof-of-concept application, demonstrating the integration of the trained model into traffic monitoring systems to enable intelligent, emergency-vehicle-aware traffic management.

5 Conclusion and Future Work

5.1 Summary of Achievements

This project addressed the critical challenge of ambulance delays in urban traffic through an AI-powered computer vision solution. We have developed and tested a YOLOv8-based vehicle detection system, leading to the following important achievements:

- **Exceptional Ambulance Detection:** Our custom-trained model yielded a mean Average Precision (mAP@0.5) of 0.780 for ambulance detection, which represents 75.5% precision and 68.8% recall, thus showing highly reliable identification of emergency vehicles.
- **Significant Performance Improvement:** This represents a 105% improvement in the overall mAP—from 0.241 for the pre-trained model to 0.496—after custom training, thus confirming the need and efficacy of domain-specific training.
- **Multi-Class Vehicle Recognition:** The system classifies five different types of vehicles: Ambulance, Bus, Car, Motorcycle, and Truck, all with good general accuracy, while providing broad capabilities for traffic analysis.

- **Real-Time Processing Capability:** Our Streamlit-based web application demonstrates real-time detection and counting on video streams, validating the system's readiness for practical deployment.
- **Optimized for practical deployment:** YOLOv8n ensures fast inference, suitable for real-time video analysis without the use of expensive specialized hardware.

Our system will not only ensure high reliability in the detection of known vehicle types but also provide a solid foundation for intelligent traffic management capable of automatically responding to emergency situations. The proposed system will reduce response times, possibly saving lives in medical emergencies with minimal misclassifications and a particularly high success rate for the critical ambulance class.

5.2 Practical Implications

This has practical importance in the successful development of such a system:

- **Emergency Response:** Quicker ambulance transit through automated traffic signal control
- **Cost-Effective Deployment:** Leverages existing traffic camera infrastructure
- **Scalability:** Can be deployed across multiple intersections within the city-wide network
- **Data-Driven Traffic Management:** Provides valuable analytics on traffic patterns and vehicle distributions

5.3 Future Work

To further enhance the system and move toward full-scale deployment, the following research directions are recommended:

1. **Improved Car Detection:** The higher false-negative rate for the Car class is because of the need for more diverse training samples, variation in lighting conditions, angles, and occlusion.
2. **Traffic Signal Control Integration:** Develop communication protocols and control algorithms necessary to interface with real-world traffic management systems and enable ”green wave” auto-generation for detected ambulances.
3. **Multi-Camera Tracking:** Object tracking across multiple camera views maintains the ambulance identity as it moves through the city, ensuring consistent priority across intersections.
4. **Real Time Alert System:** A dashboard displaying real-time alerts in traffic control centers whenever an ambulance is detected along with predicted arrival times at upcoming intersections.
5. **Extended Vehicle Classification:** Extend the system to identify other emergency vehicles such as fire trucks and police cars, and give them priority accordingly.
6. **Edge Deployment:** The model should be optimized for deployment on edge devices at intersections of traffic, reducing latency and network bandwidth.
7. **Night and Adverse Weather Performance:** Collect more training data under challenging conditions, like night, rain, and fog, for robust 24/7 operations.
8. **Explainable AI:** Develop visualization tools to understand which visual features the model uses to identify ambulances. The aim is to enhance trust and interpretability for traffic authorities.
9. **Field Testing and Validation:** Real-world pilot deployments shall be carried out to determine the actual impact on ambulance response times and system reliability in operation.

5.4 Concluding Remarks

The project shows that modern computer vision and deep learning can effectively solve critical urban infrastructure tasks. We've applied cutting-edge object detection methods, domain-specific training, and effective system design to lay the foundation for smart traffic management—intelligent and responsive to emergency vehicle priority.

While this is a very high mAP@0.5 of 0.780, the performance for ambulance detection proves that AI-powered systems can detect emergency vehicles under heavy, complex urban traffic conditions reliably. With further development and integration into traffic control infrastructures, this technology may save lives by allowing ambulances to reach patients and hospitals with no unnecessary delays.

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.