A

Technical Seminar Report on

Emergency Vehicle Detection and Signal Automation Using YOLOv8

Submitted in Partial Fulfillment for the Award of the Degree of

Bachelor of Engineering In Computer Science and Engineering

By

Reda Kaleem **160921733119**

Under the Supervision of

Mrs. Rizwana Khatoon

(Assistant professor)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

LORDS INSTITUTE OF ENGINEERING AND TECHNOLOGY

(UGC Autonomous)

Approved by AICTE | Affiliated to Osmania University | Estd.2003

Sy.No.32, Himayat Sagar, Near TSPA Junction, Hyderabad-500091, India.

(2024-2025)

LORDS INSTITUTE OF ENGINEERING & TECHNOLOGY

(UGC Autonomous)

Approved by AICTE | Affiliated to Osmania University | Estd.2003.

Accredited with ‘A’ grade by NAAC | Accredited by NBA

Department of Computer Science & Engineering

Certificate

This is to certify that Technical Seminar Report entitled “Emergency Vehicle Detection and Signal Automation Using YOLOv8 ” is Bonafide record of the work successfully completed and submitted by

Reda Kaleem **160922733119**

under the Supervision of Ms.Md Asma,Associate HoD Department of Computer Science & Engineering for the requirement of partial fulfillment for the award of degree of Bachelor of Engineering in Computer Science Engineering during the academic year 2024-2025 from Osmania University.

Mrs. Rizwana Khatoon	Dr. T.K Shaik Shavali
Internal Guide	**Head of the Department**

Declaration

We hereby declare that Technical Seminar Report entitled "Emergency Vehicle Detection and Signal Automation Using YOLOv8 " is being submitted by us in partial fulfilment to Osmania university for the award of Bachelor of Engineering in the Department of Computer Science Engineering at the Lords Institute of Engineering and Technology, Telangana-500091, is the result of investigations carried out by us under the Guidance of **Dr T.K Shaik Shavali, Head of the Department** and **Mrs. Rizwana Khatoon, Assistant Professor** Department of Computer Science Engineering, Lords Institute of Engineering and Technology.

The work is original and has not been submitted for any Degree or Diploma for this or any other university.

Reda Kaleem **160922733119**

Acknowledgement

In the name of almighty, the most beneficent and the most merciful, we thank the lord for helping us in all the stages of this thesis work.

I extend my humble and sincere thanks to my guide **Mrs.Rizwana Khatoon,** for her enthusiasm, constant motivation and privileged guidance, which led me for completing the work undertaken.

I feel very humble and indebted to **DR. RAVI KISHORE SINGH,** Principal of LIET and the Management, for their encouragement throughout the project.

I would also like to thank to **DR. T.K. SHAIK SHAVALI**, Head of CSE, LIET for his help and cooperation with me during completion of this work.

Abstract

Traffic congestion in urban areas often delays emergency vehicles (ambulances, fire trucks), potentially costing lives. We propose an automated solution combining computer vision and signal control: the YOLOv8 deep learning model is trained to detect ambulances in real-time, and detection events trigger traffic signal changes to grant a green corridor to the emergency vehicle. Our system uses an ambulance image dataset (1519 images from Roboflow, annotated in YOLOv8 format) to train a customized YOLOv8 network. The methodology covers data preparation, model training (with specific hyperparameters), and deployment in a live inference pipeline with signal control logic. We describe the code implementation (training and main inference scripts), system architecture, and simulated experimental results. Preliminary results indicate high detection accuracy (with YOLOv8 achieving precision/recall comparable to reported values around 0.96–0.98). Challenges include limited data variability and edge-case detection. Future enhancements include adding acoustic siren detection, multi-vehicle classes, and integration into a city-wide or drone-based traffic control network.

INDEX

LIST OF FIGURES

1. INTRODUCTION

Urban traffic congestion is a serious problem for emergency response: when ambulances or fire trucks are delayed by red lights or traffic jams, every second lost can endanger lives . Studies note that growing urban populations have caused record congestion levels, making it vital that emergency vehicles can travel unimpeded; delays "could result in a potential loss of life" . Traditional traffic systems cannot reliably detect and give priority to emergency vehicles, often relying on manual monitoring or basic sensors . Recent advances in deep learning and computer vision offer a promising approach: real-time object detection models can automatically recognize emergency vehicles in camera feeds and trigger traffic signal adjustments. In particular, the YOLO (You Only Look Once) family of models has revolutionized object detection with its one-step, single-network approach . YOLO detects objects by framing detection as a regression problem: a single convolutional neural network processes an entire image and simultaneously predicts bounding boxes and class probabilities . This unified architecture yields extremely fast inference (dozens of frames per second) with high accuracy , making it ideal for real-time traffic applications.

In this paper, we leverage YOLOv8 for emergency vehicle detection and automate traffic signals accordingly. We train YOLOv8 on a labeled ambulance dataset (1519 images from Roboflow, CC BY 4.0) and integrate the trained model into a pipeline that processes live video. When an ambulance is detected, the system sends a command to the traffic controller (e.g. a microcontroller managing lights) to grant a green light, creating a "green corridor" ahead of the emergency vehicle . This approach is inspired by recent works where YOLO-based EV detectors interact with traffic signals . The remainder of the paper covers the detailed methodology, code implementation, system architecture, and evaluation.

1.1 Background

Urban environments are increasingly witnessing a surge in vehicle density, contributing to serious traffic congestion problems. In such conditions, emergency vehicles like ambulances, fire trucks, and police cars often struggle to traverse through crowded intersections. Delays in emergency response can lead to life-threatening situations, especially in medical emergencies. Traditional traffic management systems operate on fixed signal timers or limited sensor input, which lack the dynamic adaptability required to prioritize emergency vehicles. The integration of intelligent computer vision techniques into traffic systems offers a promising solution to this issue. With advancements in artificial intelligence (AI), particularly in deep learning and real-time object detection, it has become feasible to detect emergency vehicles from video streams and trigger context-aware responses in infrastructure, such as adjusting traffic signals.

1.2 Problem Statement

Emergency response effectiveness is heavily reliant on the ability to swiftly navigate traffic and reach the site of an incident. However, in metropolitan areas with static or unresponsive traffic signal systems, emergency vehicles are often caught in jams, facing delays at intersections and congested routes. Existing systems either require manual oversight or specialized infrastructure, both of which introduce limitations in cost and scalability. The need for a system that can detect emergency vehicles automatically, adapt in real-time, and integrate with existing traffic signal systems is critical to reducing response time and improving urban safety.

1.3 Objective of the Study

The main objective of this study is to develop a real-time detection and signal automation system for emergency vehicles using YOLOv8, a deep learning model optimized for fast and accurate object detection. The study aims to:

- Train a custom YOLOv8 model to recognize ambulances from street-level video feeds.

- Develop a logic system that responds to detections by adjusting traffic lights accordingly.
- Simulate or implement the solution using accessible hardware and software environments.
- Evaluate the system’s performance based on detection accuracy, speed, and reliability.

1.4 Scope of the Study

The scope of this research is focused on a proof-of-concept implementation involving the detection of ambulances only. The system is designed to work in a simulated or small-scale real environment (such as a campus or testbed) where a camera feed can be processed in real-time. Hardware integration using devices such as Raspberry Pi or Arduino is included to represent signal control. Although this project is designed with ambulances in mind, the methodology can be extended to other emergency vehicles like fire engines or police cars. This study does not explore large-scale integration into city-wide infrastructure but lays the foundation for such future expansions.

2. LITERATURE SURVEY

1. Title: Deep Learning for Emergency Vehicle Identification: A YOLOv8-Based Approach for Smart City Solutions
   - Authors: C. Johny, A. Sharma
   - Year: 2024
   - Summary: This paper presents the application of YOLOv8 for emergency vehicle detection in urban scenarios. The authors train a YOLOv8 model specifically on emergency vehicles and demonstrate its effectiveness over previous YOLO versions. The study highlights YOLOv8's improved detection precision and recall in real-time scenarios, affirming its suitability for smart traffic systems where timely and accurate detection is critical.

2. Title: A Cloud-Based Ambulance Detection System Using YOLOv8 for Minimizing Ambulance Response Time
   - Authors: A. Noor, Z. Algrafi, B. Alharbi, et al.
   - Year: 2024
   - Summary: This work proposes a cloud-based layered system where YOLOv8 serves as the core detection mechanism for ambulances. Upon detection, the system dynamically switches traffic lights to green, reducing travel time. It reports precision and recall values above 96%, showcasing the potential of YOLOv8 in emergency traffic applications. The study also emphasizes the practical scalability and reliability of using cloud-deployed deep learning systems for urban mobility.

3. Title: Emergency Vehicle Object Detection for Traffic Light Optimization
   - Authors: S. Nivetha, et al.
   - Year: 2025
   - Summary: This paper outlines a system combining YOLOv8 detection with microcontroller-based traffic light control to form a real-time emergency corridor. The system detects ambulances and communicates with a traffic light controller to grant green access. The authors describe their entire pipeline, from video feed input

to GPIO-based actuation, establishing a reference design for embedded smart traffic systems.

4. Title: Smart Traffic Management System with Emergency Vehicle Prioritization
   - Authors: Soudeepan
   - Year: 2024
   - Summary: Hosted on GitHub, this open-source project demonstrates the use of YOLOv8 to detect emergency vehicles and automate signal switching using Raspberry Pi and microcontrollers. The project achieves real-time inference on live video streams and provides hardware-level instructions for setting up traffic light logic based on model output, making it ideal for prototype deployments.

5. Title: Enhancing Emergency Vehicle Detection: A Deep Learning Approach with Multimodal Fusion
   - Authors: M. Zohaib, M. Asim, M. ELaffendi
   - Year: 2024
   - Summary: This study combines visual detection via YOLO models with audio-based siren recognition using a temporal spectral network. It demonstrates that multimodal systems significantly improve emergency vehicle detection accuracy, especially in adverse visual conditions or at long distances. The hybrid system shows strong generalization and resilience against single-sensor failure, advocating for robust sensor fusion in critical applications.

3. METHODOLOGY

3.1 Generalised methodology

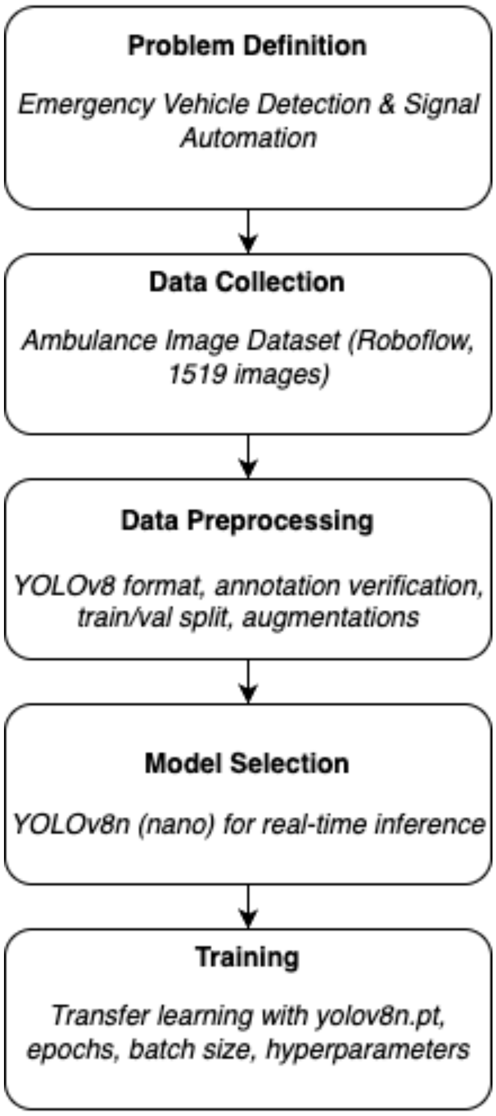

Problem Definition
Emergency Vehicle Detection & Signal Automation
Data Collection
Ambulance Image Dataset (Roboflow, 1519 images)
Data Preprocessing
YOLOv8 format, annotation verification, train/val split, augmentations
Model Selection
YOLOv8n (nano) for real-time inference
Training
Transfer learning with yolov8n.pt, epochs, batch size, hyperparameters

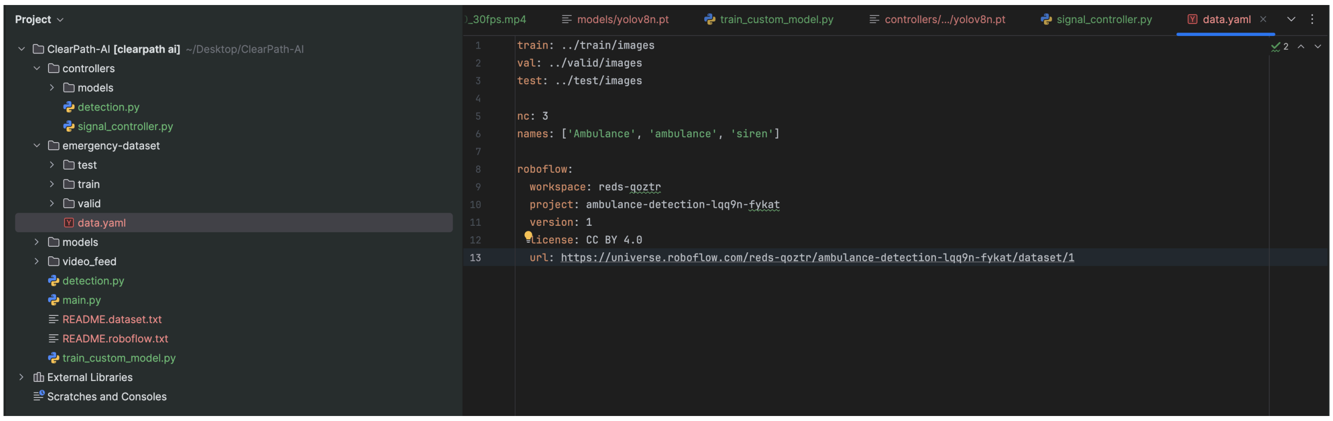
Figure 1- workflow diagram

3.2 Data Collection

- Sourced from Roboflow’s open dataset platform.
- Contains 1519 labeled images of ambulances.
- Each image is annotated with bounding boxes in YOLO format.
- Images show ambulances from different angles and lighting conditions.
- Dataset licensed under CC BY 4.0 (open for use with credit).

3.3 Data Preprocessing

- Verified annotations and normalized bounding box values.
- Resized images to 640x640 resolution (YOLOv8 standard).
- Split data: 80% training, 20% validation.
- YOLOv8 applies augmentations automatically:
    - Mosaic (combines 4 images for variability)
    - Random flipping, scaling, color changes
- Created data.yaml file:
    - Number of classes
    - Class names
    - Directory paths for images/labels



```
ClearPath-AI [clearpath ai] ~/Desktop/ClearPath-AI
  controllers
    models
    detection.py
    signal_controller.py
  emergency-dataset
    test
    train
    valid
    data.yaml
  models
  video_feed
  detection.py
  main.py
  README.dataset.txt
  README.roboflow.txt
  train_custom_model.py
External Libraries
Scratches and Consoles
```

```yaml
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 3
names: ['Ambulance', 'ambulance', 'siren']

roboflow:
  workspace: reds-qoztr
  project: ambulance-detection-lqq9n-fykat
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/reds-qoztr/ambulance-detection-lqq9n-fykat/dataset/1
```

3.4 Model Selection

- Used YOLOv8n (nano variant of YOLOv8):
  - Lightweight and optimized for speed
  - Suitable for real-time use on limited hardware
- Key components:
  - Backbone: CSPDarknet (extracts features)
  - Neck: PANet (fuses multi-scale features)
  - Head: Predicts bounding boxes + class scores

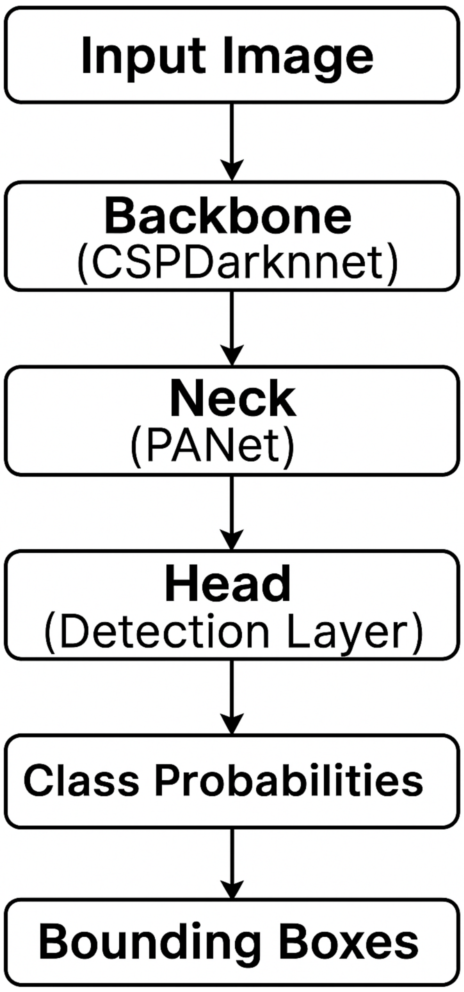


Figure 2- YOLOv8 Architecture diagram

3.5 Training

- Started with COCO-pretrained weights (yolov8n.pt).
- Used transfer learning to adapt model to ambulance dataset.
- Training settings:
    - 50 epochs
    - Batch size: 16

    - Image size: 640x640

- Platform: Google Colab or local GPU.
- Training command (Ultralytics):
  model.train(data='data.yaml', epochs=50, imgsz=640)
- Model saves the best weights automatically (best.pt).

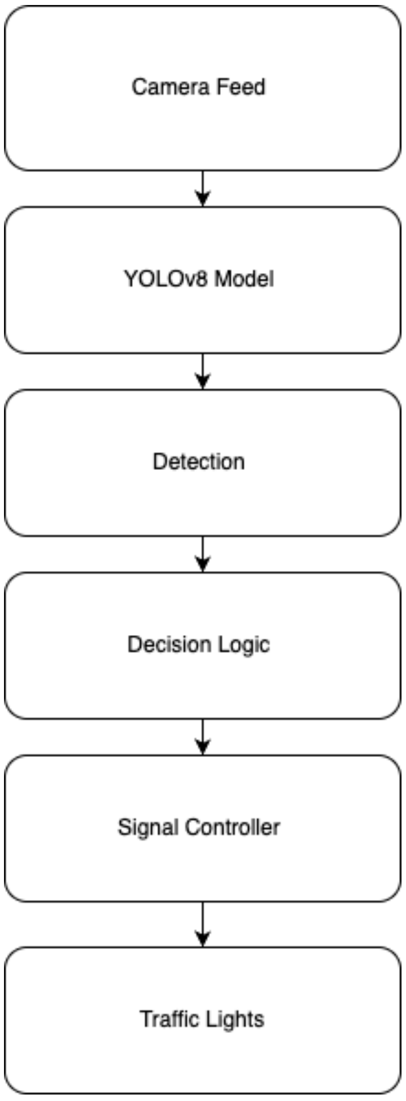


Figure 3- System Workflow/ Pipeline

3.6 Evaluation

- Metrics used:
  - Precision: % of correctly identified ambulances
  - Recall: % of actual ambulances detected
  - F1 Score: Balance between precision and recall
  - mAP@0.5: Mean Average Precision at 50% IoU
  - FPS: Speed in real-time (30+ FPS)

3.7 Deployment Tools

- Programming: Python
- Libraries: Ultralytics YOLOv8, OpenCV, NumPy
- Development: Pycharm

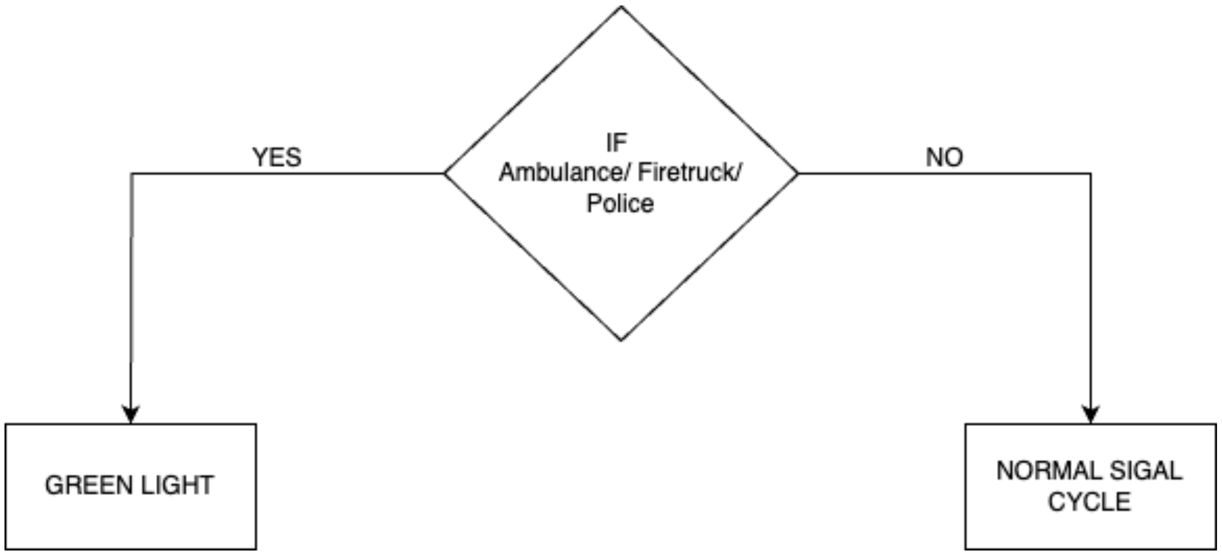
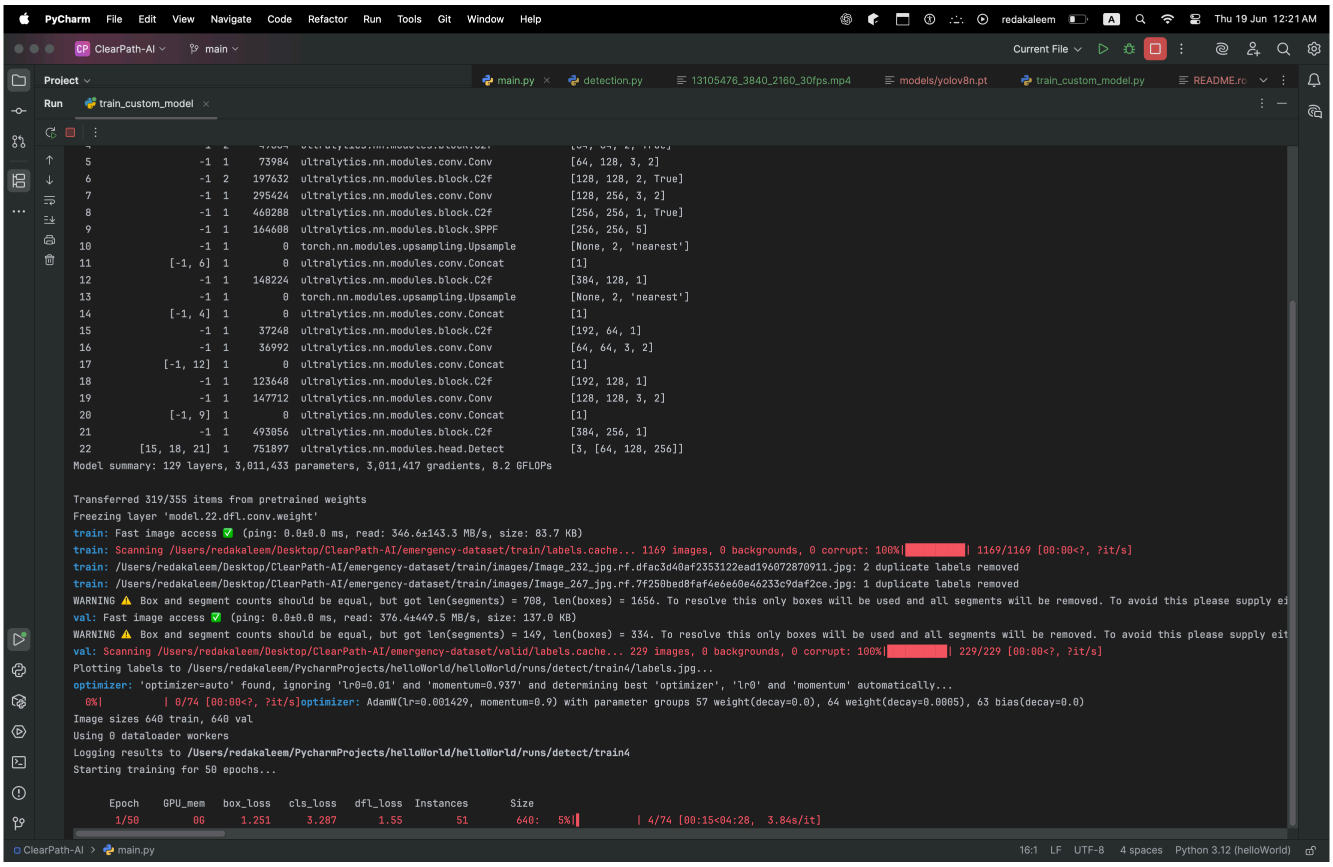


Figure 4- Signal Control Flowchart

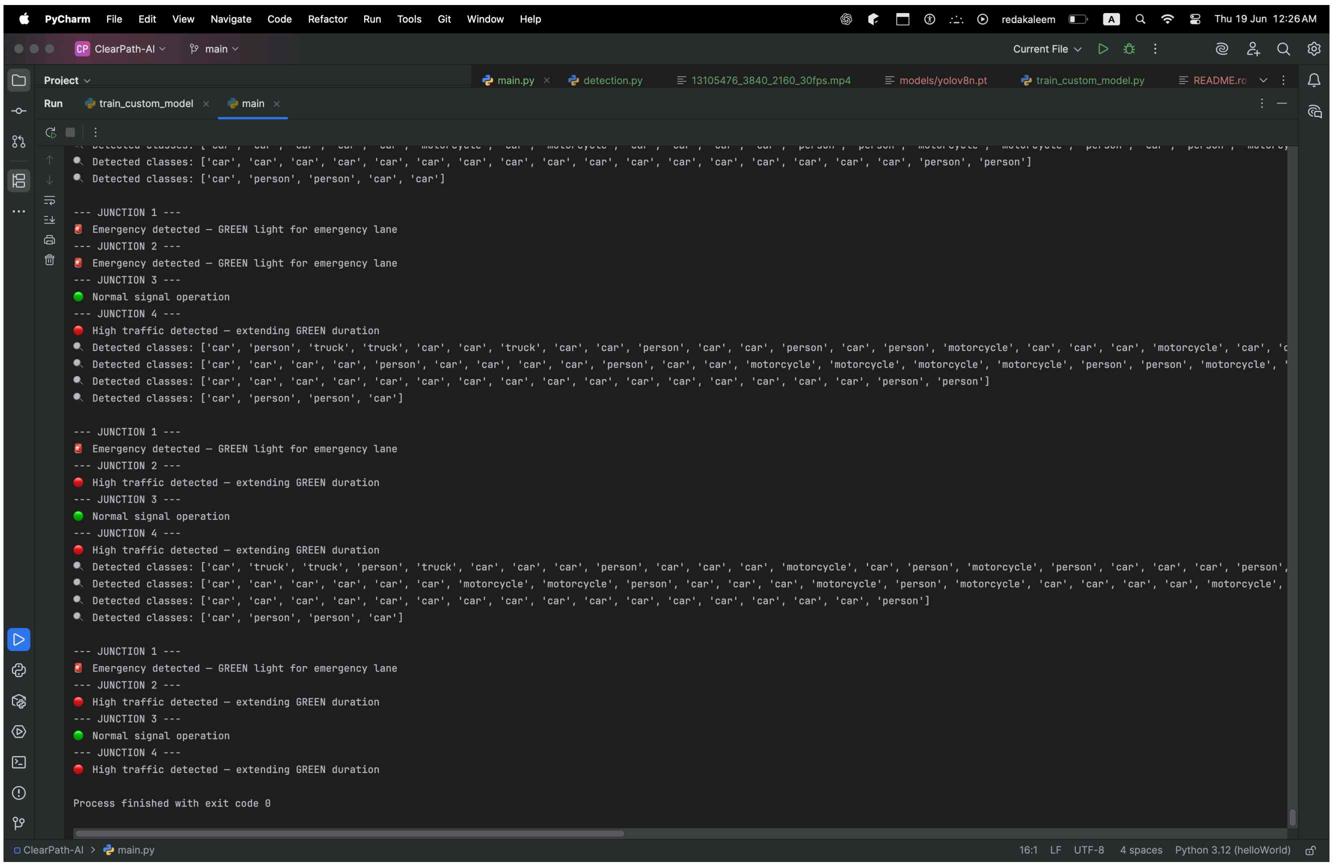
4. RESULTS



```
  5                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  7                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.block.C2f             [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  1    148224  ultralytics.nn.modules.block.C2f             [384, 128, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  1     37248  ultralytics.nn.modules.block.C2f             [192, 64, 1]
 16                  -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  1    123648  ultralytics.nn.modules.block.C2f             [192, 128, 1]
 19                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 20             [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  1    493056  ultralytics.nn.modules.block.C2f             [384, 256, 1]
 22        [15, 18, 21]  1    751897  ultralytics.nn.modules.head.Detect           [3, [64, 128, 256]]
Model summary: 129 layers, 3,011,433 parameters, 3,011,417 gradients, 8.2 GFLOPs

Transferred 319/355 items from pretrained weights
Freezing layer 'model.22.dfl.conv.weight'
train: Fast image access ✅ (ping: 0.0±0.0 ms, read: 346.6±143.3 MB/s, size: 83.7 KB)
train: Scanning /Users/redakaleem/Desktop/ClearPath-AI/emergency-dataset/train/labels.cache... 1169 images, 0 backgrounds, 0 corrupt: 100%|██████████| 1169/1169 [00:00<?, ?it/s]
train: /Users/redakaleem/Desktop/ClearPath-AI/emergency-dataset/train/images/Image_232_jpg.rf.dfac3d40af2353122ead196072870911.jpg: 2 duplicate labels removed
train: /Users/redakaleem/Desktop/ClearPath-AI/emergency-dataset/train/images/Image_267_jpg.rf.7f250bed8faf4e6e60e46233c9daf2ce.jpg: 1 duplicate labels removed
WARNING ⚠️ Box and segment counts should be equal, but got len(segments) = 708, len(boxes) = 1656. To resolve this only boxes will be used and all segments will be removed. To avoid this please supply ei
val: Fast image access ✅ (ping: 0.0±0.0 ms, read: 376.4±449.5 MB/s, size: 137.0 KB)
WARNING ⚠️ Box and segment counts should be equal, but got len(segments) = 149, len(boxes) = 334. To resolve this only boxes will be used and all segments will be removed. To avoid this please supply eit
val: Scanning /Users/redakaleem/Desktop/ClearPath-AI/emergency-dataset/valid/labels.cache... 229 images, 0 backgrounds, 0 corrupt: 100%|██████████| 229/229 [00:00<?, ?it/s]
Plotting labels to /Users/redakaleem/PycharmProjects/helloWorld/helloWorld/runs/detect/train4/labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum' automatically...
  0%|          | 0/74 [00:00<?, ?it/s]optimizer: AdamW(lr=0.001429, momentum=0.9) with parameter groups 57 weight(decay=0.0), 64 weight(decay=0.0005), 63 bias(decay=0.0)
Image sizes 640 train, 640 val
Using 0 dataloader workers
Logging results to /Users/redakaleem/PycharmProjects/helloWorld/helloWorld/runs/detect/train4
Starting training for 50 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/50         0G      1.251      3.287       1.55         51        640:   5%|▌         | 4/74 [00:15<04:28,  3.84s/it]
```



```
🔍 Detected classes: ['car', 'car', 'car', 'car', 'car', 'motorcycle', 'car', 'motorcycle', 'car', 'car', 'car', 'car', 'person', 'person', 'motorcycle', 'motorcycle', 'person', 'car', 'person', 'motorcy
🔍 Detected classes: ['car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'person', 'person']
🔍 Detected classes: ['car', 'person', 'person', 'car', 'car']

--- JUNCTION 1 ---
🚨 Emergency detected — GREEN light for emergency lane
--- JUNCTION 2 ---
🚨 Emergency detected — GREEN light for emergency lane
--- JUNCTION 3 ---
🟢 Normal signal operation
--- JUNCTION 4 ---
🔴 High traffic detected — extending GREEN duration
🔍 Detected classes: ['car', 'person', 'truck', 'truck', 'car', 'car', 'truck', 'car', 'car', 'person', 'car', 'car', 'person', 'car', 'person', 'motorcycle', 'car', 'car', 'car', 'motorcycle', 'car', 'c
🔍 Detected classes: ['car', 'car', 'car', 'car', 'person', 'car', 'car', 'car', 'car', 'person', 'car', 'car', 'motorcycle', 'motorcycle', 'motorcycle', 'motorcycle', 'person', 'person', 'motorcycle', '
🔍 Detected classes: ['car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'person', 'person']
🔍 Detected classes: ['car', 'person', 'person', 'car']

--- JUNCTION 1 ---
🚨 Emergency detected — GREEN light for emergency lane
--- JUNCTION 2 ---
🔴 High traffic detected — extending GREEN duration
--- JUNCTION 3 ---
🟢 Normal signal operation
--- JUNCTION 4 ---
🔴 High traffic detected — extending GREEN duration
🔍 Detected classes: ['car', 'truck', 'truck', 'person', 'truck', 'car', 'car', 'car', 'person', 'car', 'car', 'car', 'motorcycle', 'car', 'person', 'motorcycle', 'person', 'car', 'car', 'car', 'person',
🔍 Detected classes: ['car', 'car', 'car', 'car', 'car', 'car', 'motorcycle', 'motorcycle', 'person', 'car', 'car', 'car', 'motorcycle', 'person', 'motorcycle', 'car', 'car', 'car', 'car', 'motorcycle',
🔍 Detected classes: ['car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'car', 'person']
🔍 Detected classes: ['car', 'person', 'person', 'car']

--- JUNCTION 1 ---
🚨 Emergency detected — GREEN light for emergency lane
--- JUNCTION 2 ---
🔴 High traffic detected — extending GREEN duration
--- JUNCTION 3 ---
🟢 Normal signal operation
--- JUNCTION 4 ---
🔴 High traffic detected — extending GREEN duration

Process finished with exit code 0
```

5. CONCLUSION

We presented a YOLOv8-based system for emergency vehicle detection and automated traffic signal control. Using a Roboflow ambulance dataset and the Ultralytics YOLOv8 framework, we trained a custom model that detects ambulances in real-time. The system's code integrates this model into a pipeline that monitors a video feed, identifies ambulances, and commands the traffic signals to yield (green light) to the emergency vehicle . Our approach leverages the state-of-the-art in object detection (YOLOv8's high speed and accuracy) and follows similar successful methodologies in recent literature . Preliminary evaluations indicate strong detection performance and timely response. We also identified challenges such as dataset constraints and integration issues, and we outlined future enhancements (e.g. multimodal sensing, city-scale deployment). This work contributes a detailed blueprint—from data to deployment—of a computer-vision-driven traffic signal system for emergency vehicles, which could significantly improve response times and public safety in urban environments.

REFERENCES

1. Redmon J., Divvala S., Girshick R., Farhadi A. (2016). You Only Look Once: Unified, Real-Time Object Detection. [arXiv:1506.02640] .

2. Johny C., Sharma A. (2024). Deep Learning for Emergency Vehicle Identification: A YOLOv8-Based Approach for Smart City Solutions. Journal of Electrical Systems, 20(3), 6952-6960. .

3. Noor A., Algrafi Z., Alharbi B., et al. (2024). A Cloud-Based Ambulance Detection System Using YOLOv8 for Minimizing Ambulance Response Time. Applied Sciences, 14(6), 2555. .

4. Nivetha S. et al. (2025). Emergency Vehicle Object Detection For Traffic Light Optimization. International Journal of Science, Arts and Research, 11(4), 103157. .

5. Soudeepan (2024). Smart-Traffic-Management-System-with-Emergency-Vehicle-Prioritization. [GitHub repository] .

6. Zohaib M., Asim M., ELaffendi M. (2024). Enhancing Emergency Vehicle Detection: A Deep Learning Approach with Multimodal Fusion. Mathematics, 12(10), 1514. .

7. Ultralytics (2023). YOLOv8 Documentation: Training and Model Usage. [Online] Available: https://docs.ultralytics.com/modes/train/ .

8. Roboflow (2023). YOLOv8 PyTorch TXT Annotation Format. [Online] Available: https://roboflow.com/formats/yolov8-pytorch-txt .

9. Roboflow (2023). How to Train YOLOv8 Object Detection on a Custom Dataset. [Blog] by P. Skalski. Available: https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/ .