# Computer Vision based Dynamic Traffic Signal Duration Management using YOLOv8

# A PROJECT REPORT


*Submitted by*

**KUMAR LAXMIKANT**

**(Reg. No. CH.EN.U4CSE20039)**

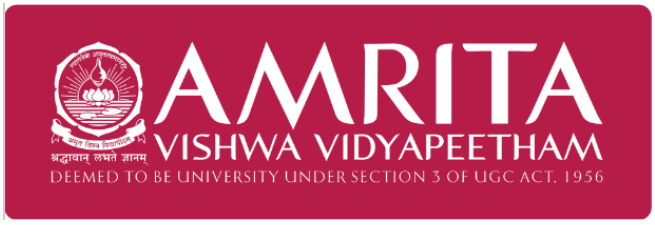

***in partial fulfillment for the award of the degree of***

**BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**

*Under the guidance of*

**Prof. Dr. M. Ravichandran**

**Submitted to**



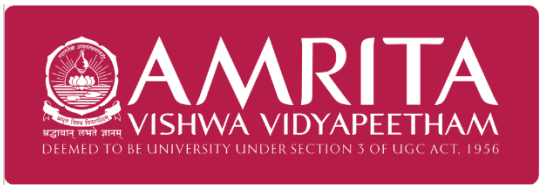


**AMRITA VISHWA VIDYAPEETHAM**

**AMRITA SCHOOL OF COMPUTING**

**CHENNAI – 601103**


**April 2024**




SCHOOL OF COMPUTING CHENNAI

# BONAFIDE CERTIFICATE

This is Certification that this project report entitled **"Computer Vision based Dynamic Traffic Signal Duration Management using YOLOv8"** is the bonafide work of **"Mr. Kumar Laxmikant (Reg. No. CH.EN.U4CSE20039)",** who carried out the project work under my supervision.

**SIGNATURE**

**Dr. S SOUNTHARRAJAN**
**Chairperson, CSE,**
Amrita School of Computing
Chennai.

**SIGNATURE**

**Dr. M. Ravichandran**
**SUPERVISOR**
Amrita School of Computing
Chennai.

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT, 1956

SCHOOL OF COMPUTING CHENNAI

# DECLARATION BY THE CANDIDATE

I declare that the report entitled **"Computer Vision based Dynamic Traffic Signal Duration Management using YOLOv8"** submitted by me for the degree of Bachelor of Technology is the record of the project work carried out by me under the guidance of **"Dr. M. Ravichandran"** and this work has not formed the basis for the award of any degree, diploma, associateship, fellowship, titled in this or any other University or other similar institution of higher learning.

**SIGNATURE**

**Kumar Laxmikant**

**(Reg. No. CH.EN.U4CSE20039)**

# ABSTRACT

In the context of burgeoning urbanization and the escalating challenges posed by the proliferation of vehicles, this research contributes to the realm of Dynamic Traffic Signal System (DTSS) by focusing on the transformative potential of YOLOv8. In this study, YOLOv8's advanced capabilities are harnessed for real-time, lane-specific vehicle density detection at traffic intersections. The innovation lies in the integration of YOLOv8's object detection prowess to discern the density of vehicles in each lane, enabling dynamic adjustments to traffic signal durations. This data-driven approach empowers the traffic management system to intelligently optimize traffic flow and minimize congestion based on observed vehicle density. By merging computer vision with traffic engineering, the paper explores the technical intricacies of implementing YOLOv8 for vehicle density estimation and emphasizes its seamless integration into traffic signal management. The results illuminate a promising trajectory towards adaptive and responsive traffic control systems, offering insights into the future of urban mobility, where traffic signals harmonize with the dynamic rhythm of city life.

**Keywords:** Urban Traffic Management, Dynamic Traffic Signal System, YOLOv8, Real-time Vehicle Detection, Computer Vision, Urban Mobility

# ACKNOWLEDGEMENT

This project work would not have been possible without the contribution of many people. It gives me immense pleasure to express my profound gratitude to our honorable Chancellor **Sri Mata Amritanandamayi Devi,** for her blessings and for being a source of inspiration. I am indebted to extend my gratitude to our **Sampoojya Swami Vinayamritananda Puri**, Administrative Director, and Shri**. I B Manikantan**, Campus Director for facilitating us all the facilities and extended support to gain valuable education and learning experience.

I register my special thanks to **Dr. V. Jayakumar,** Principal for the support given to me in the successful conduct of this project. I wish to express my sincere gratitude to **Dr. S Sountharrajan,** Chairperson, CSE, **Dr.Bhuvaneswari,** Program Head, CSE-CT and **Dr. Suthir Sriram,** Project Coordinator, CSE-CT and **Dr. M. Ravichandran,** Supervisor, CSE for their inspiring guidance, personal involvement and constant encouragement during the entire course of this work.

I am grateful to Review Panel Members and the entire faculty of the Department of Computer Science & Engineering, for their constructive criticisms and valuable suggestions which have been a rich source to improve the quality of this work.

**Kumar Laxmikant**

**(Reg. No. CH.EN.U4CSE20039)**

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF SYMBOLS AND ABBREVIATIONS

| Abbreviation | Full Form |
|---|---|
| ML | Machine Learning |
| CV | Computer Vision |
| YOLO | You Only Look Once |
| DTSS | Dynamic Traffic Signal System |
| ITS | Intelligent Traffic System |
| FPS | Frames Per Second |
| mAP | Mean Average Precision |
| CNN | Convolutional Neural Network |
| IoU | Intersection Over Union |
| AQL | Average Queue Length |
| AWT | Average Waiting Time |
| AS | Average Speed |
| EDA | Exploratory Data Analysis |
| MSCOCO | Microsoft Common Objects in Context |
| RCNN | Regions with Convolutional Neural Networks |
| CLAHE | Contrast Limited Adaptive Histogram Equalization |
| SGD | Stochastic Gradient Descent |
| GFLOPs | Giga Floating-Point Operations Per Second |
| SPP | Spatial Pyramid Pooling |

# CHAPTER 1
# INTRODUCTION

## 1.1 Project Background

Swift urbanization and the skyrocketing surge in vehicular activity pose an acute dilemma for contemporary metropolises. Pervasive gridlock, surging accident frequencies, and an ineffective traffic administration are prevalent predicaments impacting urbanites and the economic landscape alike. The aftermath is glaring—incessant traffic snarls, mounting accident rates, and an urgent demand for more effective traffic management interventions. To confront these quandaries, this investigative manuscript centers its attention on the intersection of cutting-edge Dynamic Traffic Signal Systems (DTSS).

DTSS emerges as a linchpin in grappling with the myriad challenges tied to urban traffic governance. It is within this framework that we scrutinize the revolutionary potential of deploying avant-garde computer vision methodologies to elevate traffic regulation at junctions. In particular, our study delves into the pivotal undertaking of instantaneous detection of vehicular density within traffic intersections—a foundational element in optimizing traffic signals. This data-centric methodology empowers traffic control systems with the acumen to make judicious choices concerning signal durations, culminating in enhanced vehicular flow and a reduction in obstructions.

The amalgamation of traffic engineering and cutting-edge computer vision portends a metamorphosis in urban traffic management paradigms. By harnessing data-driven revelations and contemporaneous analytics, our objective is to contribute to the creation of urban landscapes that are not only safer but also more efficient, fostering a milieu conducive to sustainable urban habitation. In the ensuing segments, we will unravel the technical intricacies of implementing our strategy and deliberate on its potential to fabricate adaptive, responsive traffic control frameworks—an insight into the imminent future of urban mobility, wherein traffic signals seamlessly conform to the pulsating rhythm of city existence.

## 1.2 Objectives

The following are the objectives of Computer Vision based Dynamic Traffic Signal Duration Management:

- **Vehicle Detection:** Vehicle Detection and Classification would be implemented with the help of the YOLOv8 model.
- **Vehicle Density Detection:** Vehicle Density Detection would be implemented by making use of vehicle counting within Region of Interest (ROI) using Point in Polygon Algorithm.
- **Real Time Lane Adaptation:** The real time lane adaptation would be implemented with the help of dynamic green signal timing formula and algorithm.
- **Dynamic Traffic Signal Control:** Create an intelligent traffic signal control algorithm that utilizes the real-time data to dynamically adjust signal durations at each intersection, prioritizing efficient traffic flow and congestion mitigation.
- **Reduction in Congestion:** Measure the system's effectiveness in reducing traffic congestion, minimizing delays, and improving overall traffic flow efficiency.

## 1.3 Scope

This project develops an advanced urban traffic management system integrating Dynamic Traffic Signal Systems (DTSS) and computer vision. Using CCTV cameras, real-time traffic data is collected and analyzed using machine learning to understand congestion and vehicle density. An intelligent traffic signal optimization algorithm adjusts signal durations based on live data, prioritizing emergency vehicles. Rigorous testing ensures real-world effectiveness, aiming to reduce congestion, enhance fuel efficiency, improve safety, and create a more adaptive traffic system.

## 1.4 Expected Result

The envisioned outcome of the project goes beyond conventional traffic control measures by seamlessly integrating YOLOv8's state-of-the-art object detection capabilities with Dynamic Traffic Signal Systems (DTSS). The primary objective is to enable the system to dynamically adjust traffic signal durations in real time, responding not only to vehicular density but also prioritizing the swift and unimpeded passage of emergency vehicles. By accurately identifying and prioritizing these vehicles, the project aims to enhance safety measures, ensuring rapid response times during critical situations, and mitigating potential risks.

In addition to its focus on emergency vehicle prioritization, the project also emphasizes the establishment of a data-driven approach. Leveraging YOLOv8's precise insights into traffic dynamics, the system intends to make intelligent decisions for optimized traffic control. This includes efficiently facilitating the movement of emergency services through traffic intersections. With a comprehensive approach to resource utilization, scalability, and adaptability, the envisioned outcome is a robust and intelligent traffic management system that significantly

improves urban mobility, enhances safety, and ensures efficient traffic flow, especially in emergency scenarios where swift and unimpeded access for emergency vehicles is paramount.

## 1.5 Project Schedule / Gantt Chart

### 1.5.1 Phase 1

TABLE 1.1. GANTT CHART FOR PHASE 1

| TASKS | STATUS | PROJECT GANTT CHART - PHASE 1 | | | | | | | | | | | | | | | | PROJECT GANTT CHART - PHASE 2 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 2023 | | | | | | | | | | | | | | | | 2024 | | | | | | | | | | | |
| | | JULY | | | | AUGUST | | | | SEPTEMBER | | | | OCTOBER | | | | NOVEMBER | | | | DECEMBER | | | | JANUARY | | | |
| | | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 |
| PHASE -1<br>INTRODUCTION<br>1. Detail discuss with supervisor about the project<br>2. Problem statement<br>3. Objectives<br>4. Scope | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| REVIEW LITERATURE WORK<br>1. Review literature (Journal and book)<br>2. Findings additional information and knowledge about project | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| METHODOLOGY<br>1. Design Algorithms<br>2. Prediction Results | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| IMPLEMENTATION PROCESS | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ANALYZE AND DISCUSSION<br>1. Analyze data collection<br>2. Discussion | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| PREPARATION AND PRESENTATION OF FINAL REPORT | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

### 1.5.2 Phase 2

TABLE 1.2. GANTT CHART FOR PHASE 2

| TASKS | STATUS | PROJECT GANTT CHART - PHASE 1 | | | | | | | | | | | | | | | | PROJECT GANTT CHART - PHASE 2 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 2023 | | | | | | | | | | | | | | | | 2024 | | | | | | | | | | | |
| | | JULY | | | | AUGUST | | | | SEPTEMBER | | | | OCTOBER | | | | NOVEMBER | | | | DECEMBER | | | | JANUARY | | | |
| | | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 |
| PHASE -2<br>INTRODUCTION<br>1. Detail discuss with supervisor about the project<br>2. Problem statement<br>3. Objectives<br>4. Scope | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| REVIEW LITERATURE WORK<br>1. Review literature (Journal and book)<br>2. Findings additional information and knowledge about project | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| METHODOLOGY<br>1. Design Algorithms<br>2. Prediction Results | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| IMPLEMENTATION PROCESS | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ANALYZE AND DISCUSSION<br>1. Analyze data collection<br>2. Discussion | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| PREPARATION AND PRESENTATION OF FINAL REPORT | PLAN<br>ACTUAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

PLAN
ACTUAL

# CHAPTER 2
# LITERATURE REVIEW

The literature survey acts as the cornerstone for this research, delivering a broad understanding of the existing knowledge domain related to urban traffic management, computer vision applications, and Dynamic Traffic Signal Systems (DTSS). This section aims to distill insights from prior research, identify gaps in current understanding, and highlight key findings that inform the research's conceptual framework and methodology.

## 2.1 Literature review based on previous research papers

***Alpatov et al.*** [1] presents an automobile identification and counting scheme designed for live traffic monitoring. This system plays a pivotal role in enhancing traffic management and safety. By employing advanced techniques and technologies, it offers a robust solution for accurately detecting and counting vehicles in real-time. The research showcases the importance of such systems in modern traffic surveillance, where the need for efficient and reliable traffic management is paramount. Their approach provides valuable insights into the development of intelligent transportation systems and contributes to the advancement of traffic monitoring technologies.

***Chen et al.*** [2] introduce an innovative approach that leverages edge computing to enhance real-time vehicle detection. Their system achieves an impressive mean Average Precision (mAP) of 86.22% and Intersection over Union (IoU) of 75.63% using YOLOv4. This high accuracy is vital for safe autonomous navigation, particularly in complex urban traffic conditions. Additionally, the authors address real-time image segmentation, essential for comprehending the driving environment. Their work emphasizes the role of edge computing in improving processing capabilities for autonomous vehicles, promising enhanced safety and efficiency in urban transportation.

The research conducted by ***Pornpanomchai et al.*** [3] focuses on a system designed for detecting and tallying vehicles from video frames. This system involves four primary segments: Image Acquisition, Image Analysis, Object Detection and Counting, and Display Result. The study aims to assess the system's usability and efficiency by testing its ability to accurately identify and count vehicles under specific conditions. Results indicate a high level of precision, with 91.98% accuracy in the afternoon and 96.35% in the evening, suggesting that the system is proficient in real-world scenarios.

***Priemer et al.*** [4] propose a decentralized approach to adaptive signalized intersection control systems leveraging Vehicle-to-Infrastructure (V2I) communication data. Their method demonstrates significant improvements, including a 5% increase in mean traffic speed and a nearly 25% reduction in mean delay within the network at high penetration rates of 100%. This highlights the potential of V2I communication and decentralized control in addressing traffic congestion.

***Cai et al.*** [5] employs Approximate Dynamic Programming (ADP) to model and optimize traffic signal control, aiming to minimize delays and enhance traffic flow efficiency. This study's noteworthy contributions encompass the application of ADP for adaptive signal timing adjustments, resulting in a remarkable 67% reduction in vehicle delays compared to fixed-time plans at a 0.5-second resolution. Moreover, operating at a 0.5-second resolution, instead of 5.0 seconds, the same ADP controller exhibits a notable 41% improvement in performance. This study introduces a foundational technique to the adaptive signalized intersection control system, offering significant potential for alleviating traffic congestion in urban areas.

***Chabchoub et al.*** [6] presents an innovative approach to traffic light control using a combination of fuzzy logic and image processing techniques. This study addresses the need for intelligent traffic control device systems that exhibit responsiveness to dynamic traffic scenarios and optimize vehicular movement efficiently. By integrating fuzzy logic, which can handle complex decision-making processes, and image processing, which enables the system to collect data from the traffic scene, the envisioned signal control system offers an intelligent and adaptive solution. Such systems have the potential to optimize vehicular flow, mitigate traffic congestion, and enhance road safety measures. The work by Chabchoub and his team contributes to the field of smart mobility solutions, offering a new direction for the development of efficient and responsive traffic light control mechanisms.

***Padilla et al.*** [7] make a significant contribution to vehicle detection with their work on "T-YOLO." This novel approach addresses the challenges of lightweight and efficient vehicle detection. T-YOLO, based on YOLO (You Only Look Once) and multi-scale convolutional neural networks (CNNs), offers a compact yet robust solution optimized for resource-efficient processing. This balance between accuracy and speed makes T-YOLO suitable for intelligent traffic systems and mobile devices. Padilla et al. report impressive performance metrics, including a Precision of 96.34%, Recall of 99.98%, mAP50 of 99.85%, and mAP95 of 99.79%, showcasing T-YOLO's practical utility in real-world vehicle detection scenarios. Their experiments using the PKLot 1 dataset, focusing on parking locations, meteorological states, and specific days, highlight their data-driven approach to model training and validation.

The paper by ***Avşar et al.*** [8] introduces a novel technique for discerning and tracing moving automobile, particularly in roundabout environments. The method brings two key innovations: firstly, it focuses exclusively on identifying moving vehicles to enhance tracking accuracy, and secondly, it employs a technique called trajectory union to minimize false track rates, thereby improving overall tracking performance. For vehicle detection, the system utilizes YOLOv4, a cutting-edge object detection model, while vehicle tracking employs either the Kalman filter or the DeepSORT algorithm. Evaluation of the proposed method against manual enumeration and benchmark assessment outcomes reveals its superiority, with significantly lower error rates. For instance, in a 20-minute video with 297 vehicles, the proposed method achieves an absolute error of only 14 vehicles, corresponding to a normalized absolute error percentage of 1.571%, compared to 33 vehicles and 3.704% error obtained by the benchmark method. The study also notes that while errors may increase with higher vehicle densities or complex roundabout layouts, the rate of error escalation remains substantially lower than the rate of vehicle count increase. Additionally, using DeepSORT for vehicle tracking results in even lower error rates, emphasizing the effectiveness of the proposed approach. Overall, this research introduces valuable insights and methodologies for enhancing vehicle detection and tracking in roundabout scenarios, with potential applications in traffic management and safety systems.

***Essa et al.*** [9] presented the Adaptive traffic signal control (ATSC) as an optimistic method to enhance the performance of controlled intersections, especially with the rise of interconnected automobiles and the presence of live data on vehicle movements. While numerous ATSC algorithms exist to optimize traffic flow, there's a notable gap in considering safety optimization. This gap likely stems from the absence of tools for real-time safety evaluation, although recent research has developed such models using dynamic traffic parameters. To bridge this gap, the study introduces a pioneering auto-didactic ATSC algorithm designed to prioritize risk mitigation optimization in real time. Trained using Reinforcement Learning (RL) and validated with live traffic data, the algorithm showcases a substantial reduction in traffic conflicts compared to traditional control systems. In contrast with conventional actuated signal control, the new algorithm slashes traffic conflicts by around 40%. Additionally, it underwent testing across different market penetration rates (MPRs) of CVs, revealing that 90% and 50% of its safety enhancements are attainable at MPR levels of 50% and 30%, respectively.

***Ge et al.*** [10] introduces Cooperative Deep Q-Learning with Q-Value Transfer (QT-CDQN) for networked signal coordination. QT-CDQN enables intersections to collaborate by sharing optimal Q-values, leading to synchronized traffic signal decisions. It uses Deep Q-Learning (DQL) with neural networks to tailor signal schedules based on immediate traffic conditions. Our

research complements this by integrating computer vision techniques, enhancing signal control's adaptability and responsiveness for more efficient transportation systems.

In the study by ***H. Song et al.*** [11], a vision-based vehicle detection and counting system for highway scenes was presented, utilizing deep learning techniques coupled with Gaussian mixture modeling and preprocessing methods. The authors employed Gaussian mixture modeling for surface extraction and segmentation, enhancing vehicle detection accuracy in conjunction with deep learning object detection. Subsequent preprocessing steps, including Gaussian filtering, MeanShift color correction, and the flooding filling algorithm for road surface separation, contributed to improved object detection by refining the background and isolating regions of interest. Additionally, the use of the ORB algorithm for feature extraction from detected vehicles yielded a remarkable mean Average Precision (mAP) value of 87.88%, demonstrating the efficacy of their approach in accurately detecting vehicles in complex highway environments. These findings underscore the potential of deep learning and preprocessing techniques for enhancing vehicle detection and counting, aligning with the objectives of our research in urban traffic management.

The research by ***Tayara et al.*** [12] featured in IEEE Access in 2018, proposes an innovative automobile tracking and enumeration system for high-fidelity aerial imagery with fine spatial resolution. Employing a Fully Convolutional Regression Network (FCRN), the study demonstrates a sophisticated approach to regression tasks, enabling precise localization and quantification of vehicles within complex aerial contexts. The reported performance metrics underscore the system's efficacy, with a recall of 90.51% indicating a high capacity to identify vehicles, a precision of 93.9% reflecting accurate classifications, and an F1 Score of 0.92 providing a balanced assessment of overall performance. These results collectively highlight the robustness of the FCRN-based system, offering a promising solution for accurate and detailed vehicle analysis in high-resolution aerial imagery.

The analysis done by ***Wang et al.*** [13] presents an analysis of five prominent deep learning algorithms for vehicle detection: Faster R-CNN, R-FCN, SSD, RetinaNet, and YOLOv3. The study evaluates these algorithms using the KITTI dataset and analyzes the results obtained. The performance metrics considered include Easy, Moderate, and Hard accuracy rates, frames per second (FPS), and the cumulative total of vehicles. Findings indicate that RetinaNet attains the highest accuracy across all difficulty levels, with Easy, Moderate, and Hard accuracy rates of 89.83%, 78.85%, and 68.73%, respectively. However, in terms of FPS, SSD outperforms the other algorithms significantly, achieving 14.15 FPS. These findings provided profound insights into advantages and drawbacks of each algorithm in the context of vehicle detection tasks.

***Gleason et al.*** [14] proposed a technique based on template matching and color information to detect vehicles. This research addresses the challenge of detecting vehicles in aerial images, a crucial task for various applications like surveillance and traffic monitoring. The combination of template matching and color information demonstrated remarkable performance, achieving a correct classification rate of 98.9% for vehicles and 61.9% for background. This impressive result effectively reduced false positives by more than half while retaining nearly all true positives, illustrating the potency of the proposed approach in accurately classifying vehicles in aerial imagery.

***Liu et al.*** [15] implemented a robust vehicle monitoring system based on the RCNN (Region-based Convolutional Neural Network) model. Notably, their RCNN model attained an impressive accuracy rate of 98.2% in classifying and identifying aerial vehicles in challenging aerial images. This high level of accuracy underscores the effectiveness of their approach, making it a significant contribution to the field of aerial image analysis and object detection.

***Kartik et al.*** [16] present an innovative approach to adaptive signalized intersection control system utilizing Vehicular Ad hoc Networks (VANETs). This work explores the integration of VANET data to optimize traffic signal timings dynamically. Their adaptive control strategy aims to enhance vehicular movement efficiency and reduce blockage. By leveraging real-time information from connected vehicles, their method demonstrates the potential to substantially boost the performance of urban traffic management systems. The study contributes valuable insights into the application of VANETs for intelligent traffic signal control, offering promising solutions for mitigating traffic-related issues in urban environments.

***Sommer et al.*** [17] conducted a comprehensive performance comparison of their proposed vehicle detection method with previously established techniques from the literature. They observed that their Faster R-CNN-based approach outperformed existing methods significantly. For instance, when comparing with the method by Liu et al. [16], which utilized ICF and an AdaBoost classifier in a soft cascade structure, the proposed Faster R-CNN achieved substantial improvements in both recall rate (85.0%) and precision rate (92.8%). This performance evaluation underscores the effectiveness and superiority of their Faster R-CNN-based automated vehicle identification system in remote sensing analysis.

***Zuraimi et al.*** [18] compares various YOLO models for vehicle detection and tracking, revealing YOLOv4 as the top performer with an impressive 82.08% mAP50 on their custom dataset. This model's success is attributed to its advanced architecture, showcasing improvements over prior iterations. YOLOv4 not only excels in accuracy but also attains an instantaneous algorithmic throughput of about 14 FPS on a GTX 1660ti graphics card, achieving a middle ground between

precision and speed. This research underscores the practical significance of YOLO models in real-time applications, particularly in domains like vehicle monitoring.

The study by ***Hassaballah et al.*** [20] introduces a holistic approach to tackle automobile surveillance and tracing under inclement weather. Their proposed visual range enhancement scheme, incorporating increased luminosity, optimization of reflective elements, and linear blending, aligns with previous research in image enhancement techniques. Leveraging a multi-scale deep convolutional neural network for robust detection and tracking echoes advancements in deep learning-based object detection methodologies. The utilization of hierarchical data associations (HDA) for track management is reminiscent of established practices in track management algorithms. The introduction of the DAWN dataset for autonomous vehicle research in adverse weather conditions is a significant contribution, facilitating benchmarking and comparison of algorithms in challenging scenarios. Performance evaluation using metrics such as MOTA and MOTP draws from established methodologies in evaluating tracking algorithms. Finally, achieving real-time tracking performance, as demonstrated by a speed of 23.25 FPS, underscores the practical applicability of the proposed approach in real-world scenarios.

***Tajalli et al.*** [21] explore urban traffic management in the context of connected and automated vehicles (CAVs). They investigate the integration of holistic speed optimization with signal control at a network level, emphasizing CAVs' transformative potential for urban mobility. By dynamically adjusting CAV speeds based on live traffic conditions and signal timing, the study aims to reduce congestion and enhance fuel efficiency. The synchronized strategy resulted in a 1.9% decrease in travel duration, a 5.3% reduction in mean latency, a 28.5% decline in the average stop count, and a 5.4% decrease in the average stop delay, when compared to scenarios exclusively optimizing signal timing parameters. The research also implements a distributed optimization algorithm in Vissim, yielding notable network performance improvements compared to single-parameter optimizations, showcasing the practicality of network-level coordination for CAVs.

***Yang et al.*** [22] made a significant stride in the domain of vehicle detection from aerial imagery, demonstrating an impressive accuracy rate of 89.44%. This remarkable achievement underscores the effectiveness of their deep learning-based approach in spot-on discernment and categorization of automobiles within aerial images, showcasing its potential for various applications, including surveillance, traffic monitoring, and urban planning.

***McKenney et al.*** [23] focus on distributed and adaptive signalized intersection control systems within realistic traffic simulations. They explore the application of distributed control

mechanisms using a sophisticated traffic simulation platform. This research contributes to the understanding of how adaptive signalized intersection control systems can effectively manage vehicular movement and congestion in complex urban environments.

The study by ***Miller et al.*** [24] presents a fresh approach using hidden Markov models (HMMs) to depict vehicle motion, offering a new solution for vehicle detection and counting challenges. Through rigorous testing on 88 hours of video data gathered from diverse locations, the proposed method displays robustness in handling variations like lighting changes, moving shadows, and camera movement. Compared to Multiple Target Tracking (MTT) and Virtual Detection Line (VDL) methods, the new approach consistently surpasses them, with the median vehicle count error being 28% lower than MTT and 70% lower than VDL. Despite the videos' low resolution (342×228 @ 30 fps) and compression to 95 kbps in h.264, reflecting practical conditions for Intelligent Transportation Systems (ITS), the proposed method shows notable performance. It achieves a median five-minute bin error of 0.0686 for the counting task, markedly outperforming both MTT and VDL, which record median errors of 0.0957 and 0.2290, respectively, on the same dataset.

***Seenouvong et al.*** [25] has developed a useful tool that uses computer vision to detect and count vehicles. Their method consists of several steps. Firstly, they distinguish the background from other objects. Then, they focus on specific areas of interest and apply visual enhancements to clarify the image. They locate each vehicle by finding the central point of every object in the view. To tally and count, they used a present digital area for detection. This tool proved to be remarkably effective with about 96% accuracy. Hence, it's a reliable and systematic solution that holds promise for overcoming challenges in vehicle detection and counting. It could also enhance smart transportation systems, especially in urban settings.

***Zarei et al.*** [26] introduce "Fast-Yolo-Rec," a noteworthy advancement in the realm of rapid vehicle detection across consecutive images. Their method uniquely integrates the advantages of YOLO-based detection with a cyclic prediction model network, addressing the imperative for swift vehicle detection in dynamic scenarios. Importantly, they employ the SSAM-YOLO framework with the SemAtt_Net backbone, achieving commendable results. Their model boasts an Average Precision of 94.06% and an impressively low Miss Rate of 0.0695, highlighting its exceptional accuracy. Strikingly, the accuracy achieved by Fast-Yolo-Rec is nearly on par with YOLOv4_Tiny, a widely recognized model, while offering a substantial advantage in terms of efficiency, with approximately 29.38% fewer parameters than YOLOv4_Tiny. This pioneering approach underscores the significance of both speed and accuracy in real-time vehicle detection

scenarios, demonstrating the potential of Fast-Yolo-Rec as a compelling solution for applications requiring rapid and precise vehicle detection across consecutive image frames.

In their recent work, ***Khalifa et al.*** [27] made a significant contribution to Intelligent Transportation Systems (ITS) by employing Convolutional Neural Networks (CNNs) for vehicle detection within vision-based systems. Notably, their research utilized the YOLOv5s architecture, achieving an impressive mean Average Precision (mAP) of 97.8% on daytime data and 95.1% on nighttime data. This showcases the robustness of their CNN-based approach under varying lighting conditions and scenarios. Their work underscores the potential of computer vision techniques in urban traffic management, providing valuable insights into the integration of vehicle detection systems with dynamic traffic signal control strategies, aligning closely with our research objectives.

The paper by ***Premaratne et al.*** [28] presents a comprehensive study on automobile tracking, discerning and enumeration methods on highways, scrutinizing both traditional image processing and deep learning-based approaches, particularly exploiting the YOLOv5 architecture in various configurations. Their investigation encompasses fundamental counting methodologies, experimental setups, and results, including lane-wise vehicle counts and performance metrics across different YOLOv5 models. Notably, all YOLOv5 models demonstrate an accuracy higher than 97.9% in the experimental test video. Furthermore, the accuracy of vehicle counting only diminishes when the frames per second (FPS) drop below 11.4%. Comparatively, traditional image processing techniques utilizing headlight pairing report accuracies of 72.4% and 96%, respectively, under different experimental conditions. These findings highlight the robust performance of YOLOv5 models and underscore the importance of maintaining adequate FPS for accurate vehicle counting.

***Razakarivony et al.*** [29] focuses on vehicle detection in aerial imagery, specifically addressing the challenging task of small target detection. This research presents a small target detection benchmark to evaluate vehicle detection algorithms operating on aerial images. The benchmark serves as a valuable resource for assessing the efficacy of diverse vehicle detection methodologies, especially in the context of detecting relatively small vehicles within large-scale aerial imagery. By providing a benchmark dataset and evaluation metrics, this work contributes to fostering the improvement of vehicle detection techniques, facilitating the creation of enhanced, precise, and dependable systems for aerial imagery analysis and applications such as surveillance, traffic monitoring, and urban planning.

***Chadwick et al.*** [30] introduced a groundbreaking method for improving distant vehicle detection by integrating radar and vision data, resulting in significant enhancements in average

precision (AP). Their evaluation on both synthetic and manually labeled datasets revealed compelling outcomes: a notable 21.9% boost in AP for small objects, elevating scores from 0.178 to 0.346 when compared to using solely RGB data. Remarkably, their fusion techniques, whether through concatenation or element-wise combination, consistently surpassed the performance of standalone RGB-based detection, showcasing the effectiveness of radar-vision fusion in addressing the challenges associated with distant objects. These findings underscore the importance of combining different sensor modalities to achieve more reliable and accurate object detection systems.

***Chen et al.*** 's [31] research introduces an innovative neural network structure, YOLO v3-live, as an evolution of YOLOv3-tiny for improved object detection. YOLO v3-live achieves 87.79% mAP precision before quantization and maintains 69.79% mAP post-quantization, with a quick identification of 28 FPS. This advancement signals progress in establishing a compromise between accuracy and real-time processing, offering promising applications in various domains.

***Smith et al.*** [32] work on the Sure Trac adaptive traffic signal control system is a notable contribution. This research introduces the Surtrac system as an innovative approach to adaptive signalized intersection control systems. The system employs advanced AI and optimization techniques to dynamically optimize signal timings in response to live traffic conditions. This pioneering work showcases the potential for improving traffic flow, reducing congestion, and enhancing urban mobility through intelligent signal control systems.

***Wu et al.*** [33] introduced the Multi-Agent Recurrent Deterministic Deep Policy Gradient (MARDDPG) algorithm for decentralized traffic signal control at multiple intersections. Drawing inspiration from the DRQN and RDPG algorithms, MARDDPG incorporates recurrent neural networks (RNNs) into the Q-network. The algorithm has been tested on SUMO Simulator with an average waiting time of 2.59 ± 0.89 and average queue length of 3.04 ± 5.19. This enhancement allows traffic lights, acting as independent agents, to make more informed decisions by considering both current and past observations. Experimental results demonstrated congestion reduction and improved traffic flow, showcasing the algorithm's potential to optimize urban mobility. Wu et al.'s research represents a significant advancement in decentralized traffic management, offering insights into the value of recurrency in enhancing traffic signal control.

The study by ***Gorodokin et al.*** [34] innovatively employs machine vision to enhance adaptive traffic light control, addressing traffic flow efficiency and congestion. By integrating YOLO v4 neural network and SORT open-source tracker, they achieve a remarkable 92% accuracy in vehicle detection and classification. This work demonstrates the practicality of combining advanced deep learning and tracking techniques for effective traffic analysis. The study

underscores the potential of data-driven strategies to revolutionize urban mobility and adaptive traffic control systems.

***Dong et al.*** [35] introduced a significant contribution to computer vision and traffic management with their lightweight vehicle detection network model based on YOLOv5. Their enhancements to the YOLOv5 architecture, including the incorporation of CBAM modules and C3Ghost and Ghost modules, result in an efficient and highly accurate vehicle detection system. Notably, their model achieved an impressive mean Average Precision at IoU (Intersection over Union) threshold of 0.5 (mAP50) of 72.4% and mAP at IoU threshold of 0.95 (mAP95) of 50.5%, showcasing its robust performance. Rigorously tested on benchmark datasets, PASCAL VOC and MS COCO, the model's versatility and adaptability are evident. In our context, this research provides a valuable reference, highlighting the potential for YOLO-based models to excel in real-time vehicle detection, an essential component of our proposed dynamic traffic signal management system. The architectural enhancements, coupled with impressive performance metrics, make this work a significant foundation for our research objectives.

***Liang et al.*** [36] introduced the Double Dueling Deep Q Network (3DQN) as a key component in their urban traffic management research. Their study showcased a remarkable 25.7% reduction in average waiting times, a significant improvement compared to fixed-time strategies. The 3DQN model consistently achieved cumulative rewards greater than −50000, highlighting its effectiveness in optimizing traffic signal control. These findings underscore the potential of adaptive traffic management systems, a concept that aligns with our research's goal to enhance urban traffic efficiency using computer vision techniques for vehicle density analysis.

***Wang et al.*** [37] introduced a novel approach to macroscopic signal management using multiagent reinforcement learning (MARL). They proposed cooperative double Q-learning (Co-DQL), a distinctive algorithm within decentralized MARL. Results demonstrated Co-DQL's superiority over existing MARL approaches, achieving an average delay timing of 36.981 ± 0.509 and a mean episode reward of -2.889 ± 0.040. These findings highlight the potential of Co-DQL in effectively reducing traffic congestion and optimizing urban traffic management, making it a valuable contribution to the domain of signalized intersection management.

***Miao et al.*** [38] contribute significantly to low-light vehicle detection with their research. Their innovative approach employs the MSR algorithm for image enhancement, coupled with tailored YOLO v3 neural network training. This adaptation achieves an impressive 93.66% accuracy, operating at a rapid 30.03 fps frame rate. Their methodology effectively addresses the challenge of nighttime detection, demonstrating the versatility of YOLO v3 architecture and its potential for real-world applications.

***Rahman et al.*** [39], a live wrong-way automobile surveillence system based on the YOLO (You Only Look Once) model and centroid tracking is introduced. This research addresses a critical safety concern on roadways, as detecting wrong-way vehicles promptly is essential to prevent accidents and ensure the safety of all road users. By leveraging YOLO's capabilities for object detection and combining it with centroid tracking, the proposed system demonstrates the potential for accurate and efficient real-time detection of wrong-way vehicles. This work contributes to the field of intelligent transportation systems and highlights the importance of advanced computer vision techniques in addressing critical safety issues on the road.

***Zheng et al.*** [40] achieved impressive performance metrics, with Precision, comprehensiveness, and quality benchmarks reaching approximately 98%, 93%, and 92%, respectively. These results highlight the high accuracy and reliability of the developed approach in detecting vehicles within high-resolution highway aerial images. Such robust performance is crucial for real-world applications, where precision and recall rates are vital for the success of traffic monitoring and management systems.

The research done by ***Sanjai et al.*** [42] underscores the pivotal role of deep learning techniques in enhancing road safety through hazard detection. With an overall Precision of 0.827 and Recall of 0.705, the model demonstrates an impressive ability to accurately identify road hazards, supported by a high mAP50 score of 0.768, indicating proficiency in object comprehension essential for road safety. However, the model's performance decreases with increasing IoU threshold, with a mAP50-95 score of 0.517. Notably, the model correctly identifies 90% of speed bumps and 57% of potholes, distinguishing them from the background. Precision and Recall curves show scores of 0.827 and 0.705, respectively, for both hazard classes. Moreover, achieving a 95% average accuracy in spotting potholes, the study illustrates the effectiveness of YOLOv8 in pothole and speed bump detection, showcasing a mAP score of 0.517 for both classes. This research underscores the potential of deep learning in revolutionizing road safety applications, offering valuable insights for intelligent transportation systems.

The paper presented by ***Sowbaranic et al.*** [43] highlights the successful implementation of their proposed model, which achieved an impressive accuracy of 88%. This accuracy metric underscores the effectiveness of their approach in accurately detecting and prioritizing emergency vehicles within complex traffic scenarios. By harnessing video and audio processing technologies, the model demonstrates robust performance in real-time traffic management, ensuring timely passage for emergency vehicles while minimizing disruption to regular traffic flow. The high accuracy achieved by the proposed model is indicative of its reliability and potential for practical deployment in urban traffic control systems. This accomplishment

represents a significant contribution to the field of intelligent transportation systems, offering a promising solution to enhance emergency response capabilities and overall traffic management efficiency in urban environments.

The research carried out by ***Tarachandy et al.*** [44] introduces a method that achieved Precision of 82.7%, Recall of 81.3%, Accuracy of 72.67%, F1-Score of 81.99%, and mAP50 of 85.3% for traffic sign detection. By integrating ridgelet filters into the feature extraction process, this approach enhances local features, improving the accuracy and robustness of detection algorithms. This research offers promising advancements in computer vision for traffic sign recognition, contributing to enhanced safety and efficiency in intelligent transportation systems.

### 2.1.1 Literature Summary Table

TABLE 2.1. LITERATURE SUMMARY

| SNo | Author | Year | Algorithm/ Technique | Dataset / Simulator | Result |
|---|---|---|---|---|---|
| 1. | Alpatov et al. | 2018 | - | - | Accuracy → 99.69% |
| 2. | Chen et al. | 2023 | YOLOv4 | Cityscapes | mAP → 86.22% |
| 3. | Pornpanomchai et al. | 2008 | - | Private Dataset | Precision (Afternoon) → 91.98%<br>Precision (Evening) → 96.35% |
| 4. | Priemer et al. | 2009 | MDP | AIMSUN NG | AWT reduced by 25% |
| 5. | Cai et al. | 2009 | ADP | OPAC | AWT reduced by 67% |
| 6. | Chabchoub et al. | 2021 | Fuzzy Logic | MATLAB | - |
| 7. | Padilla et al. | 2023 | Tiny-YOLO | PKLot | mAP50 → 99.85%<br>mAP95 → 99.79% |
| 8. | Avşar et al. | 2022 | YOLOv4, DeepSORT | - | Error (14 vehicle) → 1.571%<br>Error (33 vehicle) → 3.704% |
| 9. | Essa et al. | 2020 | RL | VISSIM | Traffic Conflict reduction → 40% |
| 10. | Ge et al. | 2019 | QT-CDQN | SUMO Simulator | AWT → 9.7 ± 0.29<br>AQL → 2.32 ± 0.02 |
| 11. | H. Song et al. | 2019 | YOLOv3 | *Private Dataset* | mAP → 87.88% |
| 12. | Tayara et al. | 2017 | FCRN | DLR Munich vehicle dataset & OIRDS dataset | Recall → 90.51%<br>Precision → 93.9%<br>F1 Score → 92% |
| 13. | Wang et al. | 2019 | Faster RCNN, R-FCN, SSD, RetinaNet, | KITTI Dataset | *Avg.Precision* \| *Easy* \| *Medium* \| *Hard*<br>*Faster RCNN* \| 81.09 \| 57.47 \| 48.37 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | and YOLOv3 | | *R-FCN* | 81.24 | 79.49 | 66.01 |
| | | | | | *SSD* | 56.63 | 45.93 | 38.91 |
| | | | | | *RetinaNet* | 89.83 | 78.85 | 68.73 |
| | | | | | *YOLOv3* | 58.56 | 45.98 | 38.23 |
| 14. | Gleason et al. | 2011 | R-CNN | *Private Dataset* | Accuracy → 98.9% | | | |
| 15. | Liu et al. | 2015 | R-CNN | *Private Dataset* | Accuracy → 98.2% | | | |
| 16. | Kartik et al. | 2013 | OAF | VANET | AWT reduced by 30% | | | |
| 17. | Sommer et al. | 2017 | Faster R-CNN | VEDAI | mAP → 76.8% | | | |
| 18. | Zuraimi et al. | 2021 | YOLOv4 | *Private Dataset* | mAP50 → 82.08% | | | |
| 19. | Hassaballah et al. | 2020 | YOLOv3, GYOLOv3 | DAWN, KITTI, and MS-COCO dataset | Tracking Speed → 23.25 FPS<br>MOTA → 76.92<br>MOTP → 92.31<br>FAF → 0.09<br>Rec → 0.92<br>Pre → 1.00 | | | |
| 20. | Tajalli et al. | 2020 | DOCA | Vissim | AWT reduced by 5.3% | | | |
| 21. | Yang et al. | 2018 | DFL-CNN | ITCVD | Accuracy → 89.44% | | | |
| 22. | McKenney et al. | 2013 | - | SUMO | Performance increase of 7.36% | | | |
| 23. | Zarei et al. | 2022 | SSAM-YOLO | CDNet2014 | Avg. Precision → 94.06% | | | |
| 24. | Miller et al. | 2015 | HMM | *Private Dataset* | Error (counting task) → 0.0686<br>Error (target tracking) → 0.0957<br>Error (VDL) → 0.2290 | | | |
| 25. | Seenouvong et al. | 2016 | OpenCV | *Private Dataset* | Accuracy → 96.85% | | | |
| 26. | Khalifa et al. | 2022 | YOLOv5s | *Private Dataset* | mAP → 97.8% | | | |
| 27. | Premaratne et al. | 2023 | YOLOv5 | COCO Dataset | Accuracy > 97.9% | | | |
| 28. | Razakarivony et al. | 2016 | Random Hough Forest | VEDAI | mAP50 → 90.5% | | | |
| 29. | Chadwick et al. | 2019 | SSD | KITTI, Private Dataset | RGB + Radar, concatenation (Generated Data by Object Size, Avg.Precision [All]) → 0.506<br>RGB + Radar, element-wise (Hand-Labeled Data by Object Size, Avg.Precision [All]) → 0.279 | | | |
| 30. | Chen et al. | 2019 | YOLOv3-tiny | *Private Dataset* | mAP50 → 87.79% | | | |
| 31. | Smith et al. | 2013 | SURTRAC | - | Traffic flow efficiency increased by | | | |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | 25-40%. |
| 32. | Wu et al. | 2020 | MARDDPG | SUMO Simulator | AWT → 2.59 ± 0.89<br>AQL → 3.04 ± 5.19 |
| 33. | Gorodokin et al. | 2021 | YOLOv4 | *Private Dataset* | Accuracy → 92% |
| 34. | Dong et al. | 2022 | YOLOv5 | PASCAL VOC, MS COCO | mAP50 → 72.4%<br>mAP95 → 50.5% |
| 35. | Liang et al. | 2019 | 3DQN | SUMO Simulator | AWT reduced by 25.7% |
| 36. | Wang et al. | 2020 | Co-DQL | SUMO Simulator | AWT → 36.981 ± 0.509 |
| 37. | Miao et al. | 2020 | YOLOv3 | *Private Dataset* | Accuracy → 93.66% |
| 38. | Rahman et al. | 2020 | YOLOv3 | *Private Dataset* | Accuracy → 99% |
| 39. | Zheng et al. | 2013 | R-CNN | HRTPO | Accuracy → 92% |
| 40. | Sanjai et al. | 2023 | YOLOv8 | Kaggle Dataset | mAP50 → 76.8%<br>mAP50-95 → 51.7%<br>Precision → 82.7%<br>Recall → 70.5% |
| 41. | Sowbaranic et al. | 2022 | YOLOv5, VGGNet | *Private Dataset* | Accuracy → 88% |
| 42. | Tarachandy et al. | 2021 | YOLOv5l, Faster RCNN | TT100K dataset | Precision → 82.7%<br>Recall → 81.3%<br>Accuracy → 72.67%<br>F1-Score → 81.99%<br>mAP50 → 85.3% |

## 2.2 Suggestions based on Literature Survey

Based on the extensive literature survey, several valuable suggestions and insights emerge to inform the development of our dynamic traffic signal management system:

1. **Leverage Efficient Vehicle Detection Models:** Consider adopting efficient vehicle detection models such as T-YOLO introduced by Padilla et al. [7] or Fast-Yolo-Rec by Zarei et al. [26]. These models establish a compromise between accuracy and processing speed, making them suitable for real-time traffic analysis.

2. **Prioritize YOLO Models for Accuracy and Speed:** The studies by Zuraimi et al. [18] and Dong et al. [35] showcase the significance of YOLO models, particularly YOLOv4 and YOLOv5, in achieving high accuracy and real-time processing capabilities. These models should be explored for their potential to enhance vehicle detection.

3. **Explore Cooperative Algorithms:** Investigate the potential of cooperative algorithms for traffic signal control, as demonstrated by Ge et al. [10] and Wang et al. [37]. Cooperative reinforcement learning and decentralized signal control strategies can optimize traffic flow efficiently.

4. **Adapt to Challenging Lighting Conditions:** Learn from the research of Miao et al. [38] and Khalifa et al. [27] to address challenging lighting conditions for vehicle detection. Adaptive algorithms and preprocessing techniques can improve accuracy in scenarios with varying light levels.

5. **Incorporate Adaptive Traffic Management:** Inspired by Liang et al.'s [36] work, consider the implementation of responsive traffic management solutions. These systems can dynamically adjust signal timings based on live vehicle density analysis, optimizing urban traffic efficiency.

# CHAPTER 3
# PROBLEM STATEMENT AND METHODOLOGY

## 3.1 Problem Statement

Urban areas frequently grapple with the pervasive issue of traffic congestion, manifesting as extended travel times, fuel inefficiency, and a compromised overall transportation system efficiency. The conventional traffic signal systems, often governed by fixed time intervals, fall short in addressing the dynamic and ever-changing nature of urban traffic patterns. This limitation gives rise to persistent traffic bottlenecks and substantial delays for commuters.

The crux of the problem lies in the inherent inflexibility of traditional traffic signal systems. These systems rely on predetermined time schedules, failing to adapt swiftly to fluctuating traffic volumes and emerging congestion hotspots. As a result, they contribute to increased fuel consumption, elevated greenhouse gas emissions, and heightened frustration among commuters.

In this context, it becomes evident that a pressing need exists for an innovative traffic management approach capable of dynamically responding to live traffic conditions. This research seeks to deal with this pressing problem by exploring the fusion of Intelligent Traffic Systems (ITS) and advanced computer vision techniques to create a traffic signal management system that adapts intelligently to urban traffic dynamics. The goal is to alleviate traffic congestion, optimize travel times, enhance fuel efficiency, and ultimately usher in a more sustainable and efficient urban transportation ecosystem.

## 3.2 Methodology

The proposed methodology employs a structured flow of execution. It begins with the capture of real-time images from CCTV cameras at traffic signal intersections. These images are then processed using computer vision's object detection technique, facilitated by advanced vehicle detection models, to identify vehicles and calculate real-time traffic density. Subsequently, the traffic density data is transmitted to a central server for analysis. Within this server, sophisticated algorithms process the data to ascertain the most efficient durations for green signal allocation in each lane, taking into account the current traffic conditions. These calculated signal timings are then communicated back to the traffic signal controllers, which promptly update the signal timers accordingly. This dynamic approach enables real-time adaptation of traffic signals, effectively

optimizing traffic flow, reducing congestion, and accommodating the presence of connected and automated vehicles (CAVs) and emergency vehicles. For a visual representation of this process, refer to Figure 3.1, which provides an abstract overview of the project's execution flow.

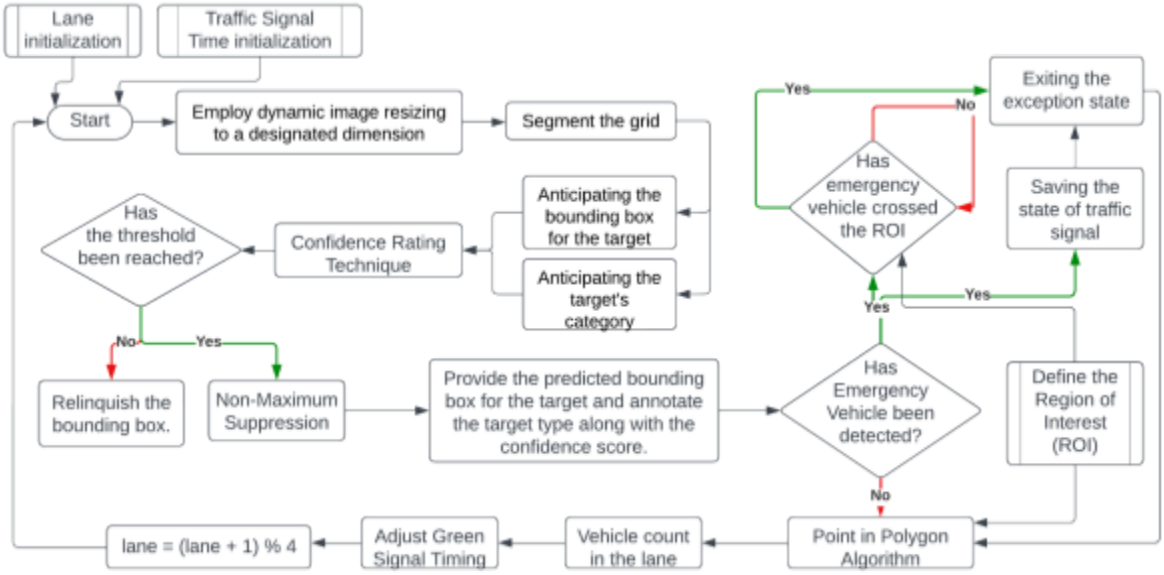



Fig 3.1. Execution Flow of the Project

## 3.3 Object Detection

Fig 3.2. Depiction of Object Detection

Object recognition stands as a crucial computer vision objective, entailing the automated recognition and accurate assessment of entities within images or video frames. This task is instrumental in numerous applications, from autonomous vehicles and surveillance systems to medical imaging and retail analytics. Modern object detection relies heavily on deep learning techniques, particularly Convolutional Neural Networks (CNNs), for precise localization & accurate classification of objects. Object detection's versatility extends to diverse domains, enabling applications such as autonomous navigation, security surveillance, medical diagnostics, and traffic management, where it plays a vital role in optimizing urban mobility and safety through real-time traffic analysis and signal control.

### 3.3.1 YOLO

The You Only Look Once (YOLO) methodology presents a neural network that concurrently anticipates bounding boxes and class probabilities. This distinguishes it from earlier object detection algorithms that repurposed classifiers for similar tasks. This unique methodology has propelled YOLO to attain remarkable outcomes, surpassing alternative real-time object detection algorithms by a considerable margin. As opposed to approaches like Faster RCNN, which employ Region Proposal Networks to discern candidate regions of interest before conducting independent perception on those regions, YOLO consolidates its predictions by unifying them through a singular fully connected layer. While methodologies utilizing Region Proposal Networks require numerous iterations for a given image, YOLO achieves its goals in a singular pass. Since its original release in 2015, various enhanced versions of YOLO have emerged, each building upon and refining its predecessor. The following timeline offers a glimpse into the ongoing development and advancements in YOLO's capabilities in recent years.

### 3.3.2 Performance evaluation metrics

For assessing & contrasting predictive proficiency of various target identification frameworks, it is essential to rely on established quantitative criteria. Among the most prevalent evaluation metrics, we commonly employ Intersection over Union (IoU) and the Mean Average Precision (mAP) metrics.

#### 3.3.2.1 Intersection Over Union (IoU)

Intersection over Union (IoU) serves as a widely used metric for assessing the precision of localization and quantifying localization discrepancies within target identification framework models. To calculate the Intersection over Union (IoU) between predicted and actual bounding boxes, the initial phase entails evaluating the shared area between the respective bounding boxes associated with a given object. Subsequently, we determine the cumulative region encompassed by these two bounding regions, designated as the "Union," denoting their collective area, and the overlapping region termed the "Intersection". By dividing the intersection by the Union, we obtain a fraction that reflects the extent of confluence relative to the overall area, offering a valuable measure of how closely the predicted bounding region aligns with the

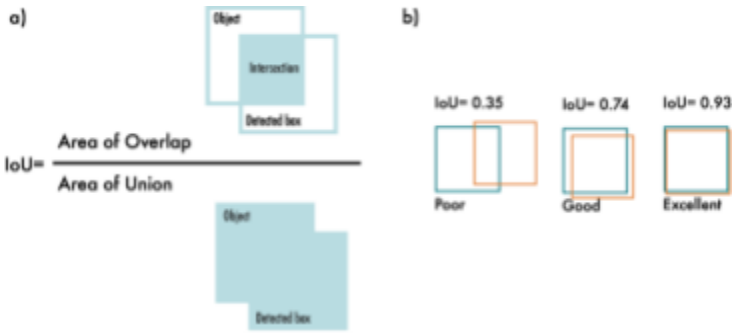



Fig 3.3. Intersection Over Union

original bounding region. In object detection, precision and recall are used for bounding region predictions rather than class predictions. A positive prediction is indicated by an IoU (Intersection over Union) value > 0.5, while a value < 0.5 signifies a negative prediction.

#### 3.3.2.2 Mean Average Precision (mAP)

The Average Precision (AP) is determined by computing the integral of the curve that depicts precision against recall for a given array of prognostications. Recall is found by dividing the total inferences generated by the model for a specific class by the total existing labels for that class. Precision, on the other hand, is the true positive rate to the entirety of model predictions, often denoted as the "true positive rate".

Precision and recall exhibit a trade-off, which is graphically illustrated by adjusting the classification threshold. The Average Precision for a model per class is obtained by calculating the area under the precision vs. recall curve. The mean Average Precision (mAP) is then computed as the average of these per-class values across all classes.

| | |
|---|---|
| $mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k = \frac{1}{n}\sum_{k=1}^{k=n} \frac{\lvert TP_k \rvert}{\lvert FP_k \rvert + \lvert TP_k \rvert}$ | $AP_k$ → The avg.precision of class k<br>n → The number of classes<br>TP → True Positive<br>FP → False Positive |

## 3.4 YOLOv8

YOLOv8 incorporates a modified CSPDarknet53 architecture as its backbone, leveraging 53 convolutional layers and implementing interconnected pathways across stages to optimize information exchange between layers. The intricate design of the CSPDarknet53 architecture serves as a robust foundation, enabling the model to capture and process intricate features within images effectively. In the head of YOLOv8, a series of convolutional layers, coupled with fully connected layers, assumes the pivotal role of predicting essential parameters such as bounding boxes, objectness scores, and class probabilities for objects recognized in an image. This intricate architecture and the utilization of cross-stage partial connections underscore YOLOv8's commitment to optimizing information extraction and representation.

An exceptional characteristic of YOLOv8 resides in its integration of a self-attention mechanism within the head of the network. This mechanism enables the model to selectively concentrate on specific regions of the image, dynamically adjusting the importance of multiple attributes based on their relevance to the assigned task. Notably, YOLOv8 stands out for its proficiency in

multi-scaled object detection, a feat achieved through the incorporation of a feature pyramid network. Comprising multiple layers, this network adeptly detects objects of diverse sizes and scales within an image, ensuring the model's versatility in accurately identifying both large and small objects across varying contexts and scenarios.

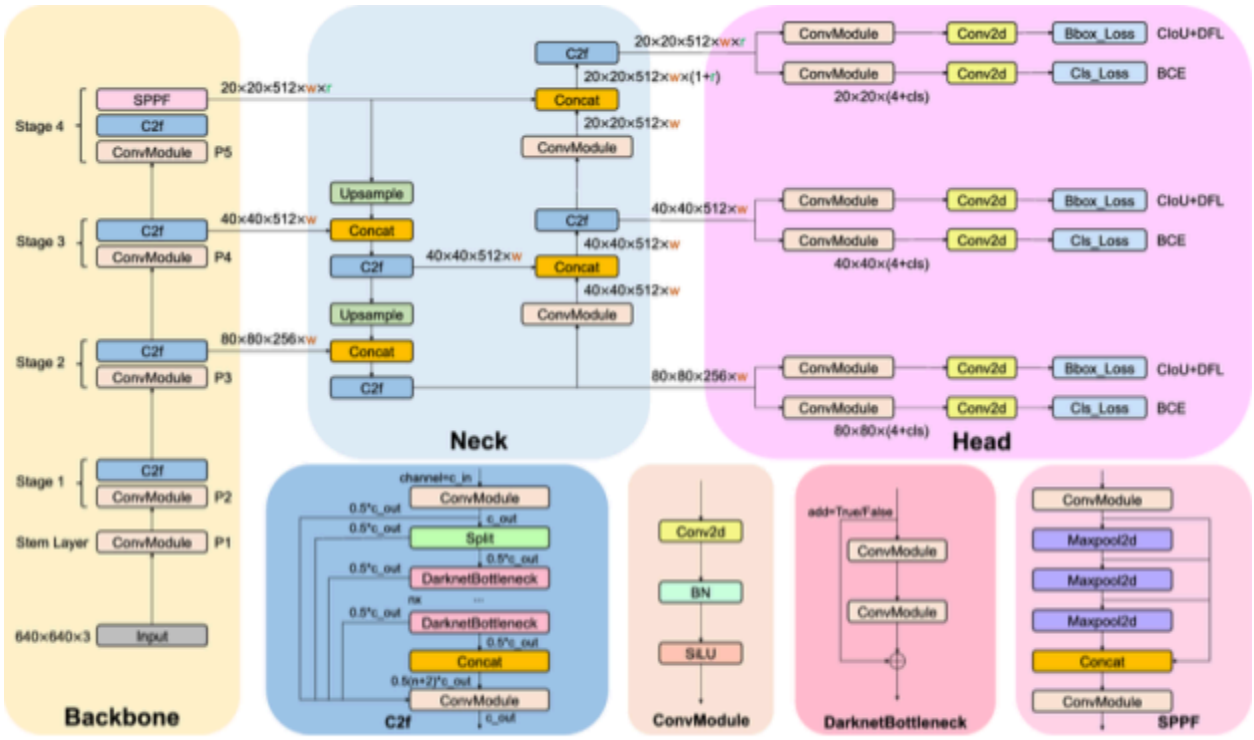



Fig 3.4. YOLOv8 Architecture

The YOLOv8 employs a convolutional neural network which is segmented into three key components: backbone, neck and head.

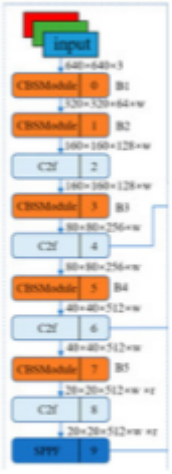



Fig 3.5. Backbone

### 3.4.1 Backbone

The backbone is the fundamental part of the neural network responsible for deriving nested characteristics from the input image. It typically consists of a series of convolutional layers, pooling layers, and other operations that incrementally reduces the geometric aspects of the input while capturing increasingly abstract and complex features. A robust backbone is crucial for the network's ability to understand and represent the content of the input image. In YOLOv8, the backbone is responsible for extracting features from the input image before they are processed further.

### 3.4.2 Neck

The neck is an intermediate part of the network situated between the backbone and the head. It is tasked with further refining and manipulating the attributes extracted by the backbone. The neck may include additional convolutional layers, skip connections, or other architectural components designed to enhance feature representation. In the context of YOLOv8, the neck plays a role in improving the spatial information flow and addressing issues such as the vanishing gradient problem. The Cross Stage Partial Network (CSPNet) is an example of a neck used in YOLOv8.

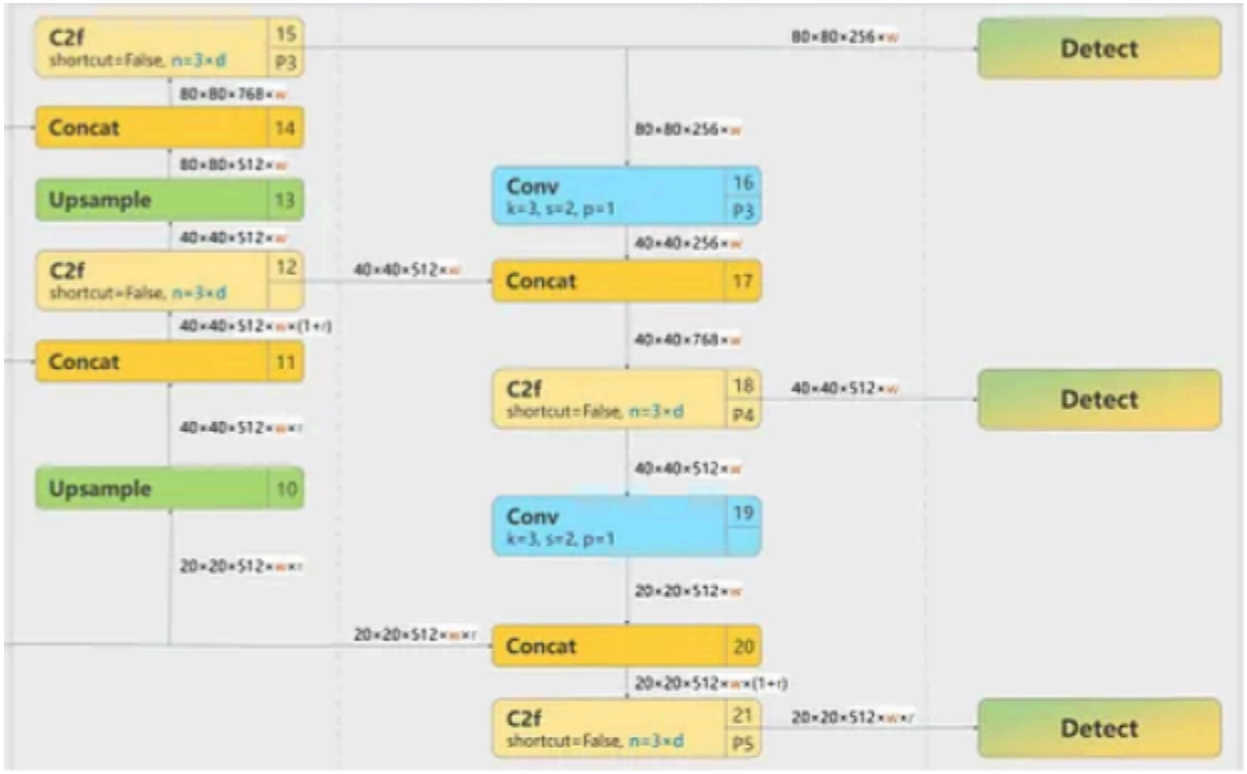



Fig 3.6. YOLOv8 Neck

### 3.4.3 Head

The head functions as the ultimate component of the network, tasked with generating predictions derived from the extracted features and refined by the backbone and neck. In the case of YOLO, the head includes the detection layers that generate bounding box coordinates, class probabilities, and objectness scores for every entity present in the input image. The head takes the high-level features from the neck and processes them to produce the final output, which represents the detected objects and their associated information.

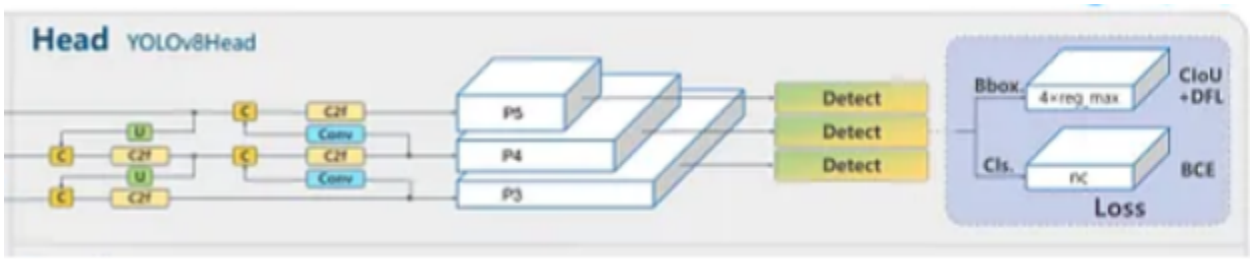



Fig 3.7. YOLOv8 Head

### 3.4.4 YOLOv8 Architecture Layers

Here is a general description of some key players in YOLOv8, based on the YOLO architecture principles:

- **Convolutional Layer (Conv):** The Convolutional layers are fundamental building blocks in neural networks, responsible for learning features from the input data.
- **CSP2F Layer (CSP2F):** The CSP2F layer is part of YOLOv8's CSPNet (Cross Stage Partial Network). CSPNet is designed to improve information flow and mitigate the vanishing gradient problem by splitting the feature map into two parts, processing each separately, and then merging them.
- **Spatial Pyramid Pooling (SPPF):** Spatial Pyramid Pooling is a technique that captures information at multiple scales by pooling features from different sub-regions of the input feature map. It facilitates the model to handle objects of various sizes.
- **YOLO Convolutional Layer (C):** The YOLO Convolutional layer is responsible for predicting bounding box coordinates, class probabilities, and objectness scores for the objects present in the grid cell.
- **Upsample Layer (U):** The Upsample layer increases the spatial resolution of the feature map, usually by duplicating rows and columns. It is used to merge features from lower-resolution layers with higher-resolution layers.
- **PANet Layer (P3, P4, P5):** PANet (Path Aggregation Network) is a feature pyramid network that enhances the flow of information across different scales. P3, P4, and P5 represent feature maps at different resolutions.
- **Detection Layer (Detect):** The Detection layer combines predictions from different scales and outputs the final set of bounding boxes, class probabilities, and objectness scores. This layer is responsible for generating the final detection results.

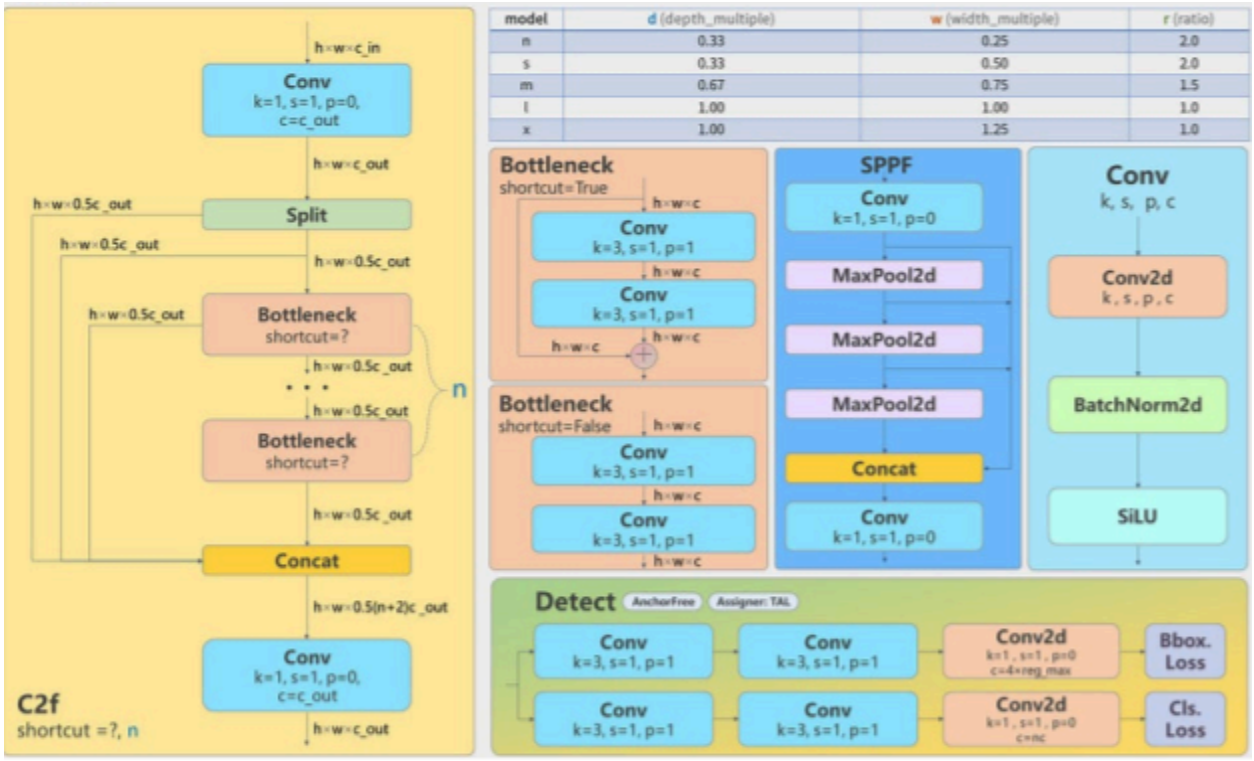



Fig 3.8. Different layers in YOLOv8 architecture

## 3.5 Point In Polygon (PIP) Algorithm

The Point In Polygon (PIP) algorithm is a computational method used to ascertain if a given point's position is interior or exterior of a closed polygon. This algorithm has various applications, including computer graphics, geographical information systems (GIS), and computational geometry.

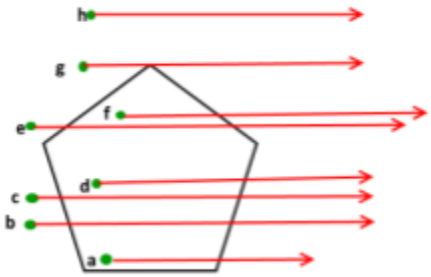



Fig 3.9. Representation of PIP Algorithm

To determine if a bounding box from a YOLO model is within a Region of Interest (ROI), a Point In Polygon (PIP) algorithm is employed. Extracting the bounding box coordinates from the YOLO output, the algorithm checks whether each corner of the bounding box lies inside the polygon defining the ROI. Utilizing a PIP algorithm like the Ray Casting or Winding Number method, the code assesses if the corners are within the ROI polygon. If all corners are inside, the bounding box is completely within the ROI; if any corner is outside, the bounding box is either partially or entirely outside the ROI. This approach ensures precise spatial relationships between bounding boxes and ROIs, facilitating accurate analysis and decision-making in applications such as object detection and tracking.

### 3.5.1 Algorithm

Given a point $(x, y)$ and a polygon with vertices $P_1$, $P_2$, ...., $P_n$, the algorithm checks if the point is inside the polygon. For each edge ($P_i$, $P_{i+1}$) in the polygon:

1. Calculate the cross product $cross_i$ between vectors $(P_i - (x, y))$ & $(P_{i+1} - (x, y))$.
2. If $cross_i > 0$ and $y$ is between $y_i$ and $y_{i+1}$, or if $cross_i < 0$ and $y$ is between $y_{i+1}$ and $y_i$, increment a counter.
3. The counter follows an odd-even rule for determining point inclusion within the polygon.

## 3.6 Dynamic Traffic Signal Duration Management

Dynamic traffic signal management, as opposed to fixed-time systems, adjusts live signal schedules based on live traffic conditions. It leverages data-driven insights like vehicle density and traffic volume to enhance traffic efficiency and alleviate gridlock. In our research, YOLOv8 is integrated for live assessment of vehicular concentration to enhance urban traffic control.

### 3.6.1 Four Phase Cycle

The four-phase traffic signal cycle represents a fundamental approach to traffic signal control, particularly at intersections. In this cycle, traffic movements are organized into four distinct phases, each with its own dedicated green signal interval as shown in Figure 3.4.

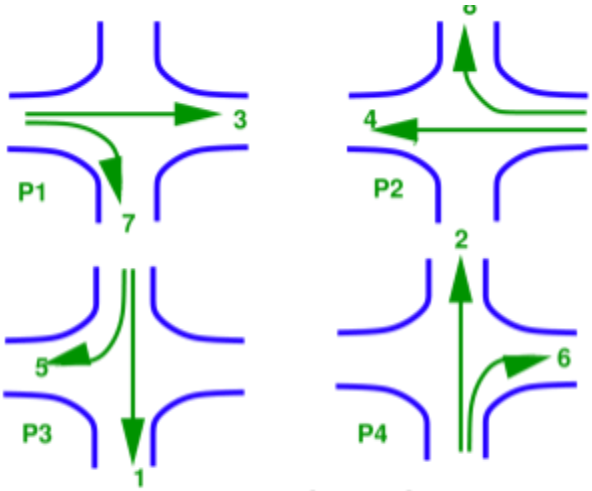



Fig 3.10. Four Phase Cycle of Traffic Intersection

For four-phase cycle we have some important terms:

#### 3.6.1.1 Cycle Time

Cycle time (as shown in Figure 3.5) is the duration it takes for a traffic signal to accomplish a full iteration operational cycle, denoted as "c." It encompasses all signal phases and intervals and is vital for optimizing traffic flow and intersection efficiency. Traffic engineers adjust cycle times to match traffic conditions.

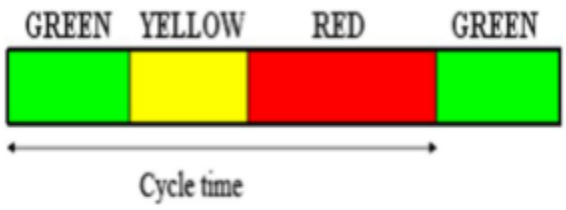



Fig 3.11. Cycle Time

### 3.6.1.2 Headway

Headway, expressed as the inter-vehicle time passing a specific point on the road, is a critical parameter in traffic management. The initial headway is determined as the time span from green signal initiation to vehicle crossing the sidewalk. The subsequent headway is characterized as the temporal interval between the passage of successive vehicles over the sidewalk. The successive headways can be plotted as shown in Figure 3.6. Once the headway stabilizes, it is referred to as Saturation headway (denoted by "h").

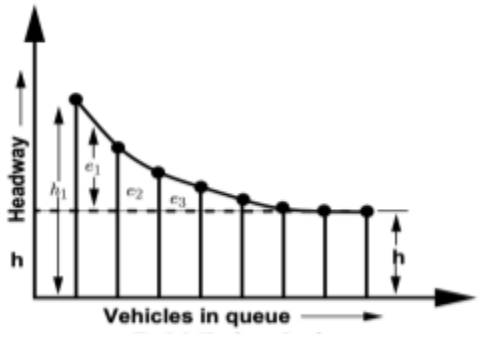


Fig 3.12. Headway Graph

If every vehicle needs *h seconds* of green time and if signals were always green, then *s vehicles/hr* would pass the intersection.

| | |
|---|---|
| $s = \frac{3600}{h}$ (Equation 1) | s → saturation flow rate in vehicles per hour of green time per lane<br>h → saturation headway in seconds |

Then, the Start-up loss time is:

| | |
|---|---|
| $L_i = \sum_{j=1}^{n} e_j$ (Equation 2) | e → Difference between actual headway and *h*'s for the $i^{th}$ vehicle |

And, the green time required to clear *N* vehicles can be found out as:

| | |
|---|---|
| $T = L_i + h * N$ (Equation 3) | T → Green time required to clear *N* vehicles |

### 3.6.1.3 Green Signal Time

To determine the ideal green signal duration determined by the number of automobiles in each category at a signalized intersection, we utilized the initial speeds and acceleration periods of vehicles. This information enabled us to estimate the mean time for automobiles in each class to cross the intersection. The computation of the green signal time is subsequently derived through the application of the following formula:

$$GST = \frac{\sum_{vehicleClass} (NoOfVehicles_{vehicleClass} * AverageTime_{vehicleClass})}{NoOfLanes}$$

GST → is green signal time

noOfVehiclesOfClass → is the count of automobiles within each vehicle class at the signal, as identified by the vehicle detection module

averageTimeOfClass → represents the mean duration required for automobiles belonging to that category to traverse an intersection

noOfLanes → denotes the total count of lanes at the intersection

The optimal traversal time for each vehicle class at an intersection can be configured based on various criteria, such as geographic region, city boundaries, specific localities, or even the unique characteristics of each intersection. This tailored approach enhances the efficacy of traffic control. Analysis of relevant data provided by transportation authorities facilitates this customization.

The signal transitions occur in a sequential manner rather than prioritizing the direction with the highest density. This aligns with the existing system, where signals sequentially transition to green without requiring individuals to adjust their behavior or causing confusion. The signal sequence remains consistent with the current system, and considerations have been made for the inclusion of yellow signals.

**Order of signals: Red → Green → Yellow → Red**

# CHAPTER 4
# EXPERIMENTAL WORK

In this section, we move from theory to practice as we implement and evaluate our proposed dynamic traffic signal management system. Through a combination of real-time CCTV camera feeds and carefully designed simulations, we aim to demonstrate the effectiveness of our approach in optimizing traffic signal durations based on vehicle density. This section outlines our methodology, datasets, and tools, offering insights into the technical aspects of our research and showcasing its real-world applicability.

## 4.1 System Requirements

### 4.1.1 Software Requirements

Listed below are the software requirements for performing project on *Machine Vision based Dynamic Traffic Signal Duration Management and Synchronization* for training & validating computer vision model and simulating the dynamic traffic intersection model:

#### 4.1.1.1 Computer Vision Model

1. **Operating System:** Operating system acts as the interface between the user programs and the kernel. Windows 8 and above (64 bit) operating system is required or macOS Catalina is required.
2. **Python Kernel:** Python stands out as a versatile and user-friendly programming language, valued for its straightforwardness and clear syntax. The requisite Python version for this context is 3.11.1.
3. **Google Colab:** Google Colab, abbreviated from Google Colaboratory, is a complimentary cloud-centric platform by Google, delivering a Jupyter Notebook atmosphere tailored for executing Python scripts.
4. **Anaconda:** Anaconda represents a freely accessible distribution of Python and R programming languages tailored for scientific computing. Its primary objective is to streamline the processes of package management and deployment, with version control facilitated through the conda package management system.
5. **Jupyter Notebook:** The Jupyter Notebook serves as an open-source web platform facilitating the development and dissemination of documents incorporating executable code, mathematical expressions, visualizations, and explanatory text. Functions involve activities like data purification and conversion, numerical simulations, statistical

modeling, data representation, machine learning, and a diverse range of other computational activities.

#### 4.1.1.2 Simulating Traffic

1. **Python Kernel:** Python is a high-level, multifaceted programming language known for its ease and comprehensibility. Python version 3.11.1 is required.
2. **Pygame module:** Pygame is a popular Python library crafted for production of 2D games and multimedia applications. It provides developers with the tools and functions needed to develop interactive and graphical applications, making it an excellent choice for hobbyist game development, educational projects, and even prototyping. Pygame version 2.5.2 is required.

### 4.1.2 Hardware Requirements

- Processor: Intel i5 2.5GHz upto 3.5GHz (or AMD equivalent)
- GPU (preferred): devoted GPU from NVIDIA or AMD with 4GB VRAM
- Memory: minimum 8GB RAM
- Secondary Storage: minimum 128GB SSD or HDD
- Network Connection: bandwidth ~ 10 Mbps to 75 Mbps

## 4.2 Dataset

The Multi-view Traffic Intersection Dataset, abbreviated as MTID [19], acts as a central resource in the domain of traffic surveillance & analysis. This dataset distinguishes itself by capturing the same traffic intersection from two distinct viewpoints shown in Fig 4.1, providing invaluable insights into traffic behavior and vehicle movements. Recognizing the fundamental importance of camera positioning in traffic surveillance, MTID offers synchronized data from two vantage points: one from a camera affixed on existing facilities and the other from a drone. This dataset encompasses 3100 frames from each viewpoint, yielding a total of 6200 frames meticulously annotated with 18883 specific labels for the infrastructure perspective and 50274 specific labels for the drone perspective as


Fig 4.1. A Traffic Intersection equipped with two Different Capturing viewpoints by mounting a camera in existing infrastructure and using a Drone equipped with a Camera

shown in Table . These annotations, performed with pixel-level precision, include axis-aligned bounding boxes and pixel-level masks, allowing for precise instance segmentation. Such comprehensive pixel-wise labels are imperative for accurate analysis, especially in scenarios where the exact positions of road users are critical, such as collision analysis.

The annotation scheme within the MTID dataset adheres to the COCO format and classes, encompassing four distinct annotations: bicycle, car, bus, and lorry. These annotations facilitate the identification and tracking of various vehicle types and bicycles within the traffic scenes. The dataset's data collection occurred at an intersection situated in downtown Aalborg, Denmark, employing two different cameras. The infrastructure viewpoint utilized an AXIS M1124-E camera with VGA resolution at 30 FPS, while the drone perspective employed a DJI Mavic Pro Drone with its standard on-board camera, capturing footage in full HD resolution at 30 FPS.

TABLE 4.1. OVERVIEW OF THE DATASET

| | **Infrastructure Viewpoint** | **Drone Viewpoint** |
|---|---|---|
| **Resolution** | 640x480 | 1920x1080 |
| **Number of frames** | 3100 | 3100 |
| **Bicycles Annotated** | 2003 | 2909 |
| **Cars Annotated** | 9989 | 32550 |
| **Buses Annotated** | 2113 | 3607 |
| **Lorries Annotated** | 4778 | 11208 |
| **Total Annotations** | 18883 | 50274 |

## 4.3 Data Preprocessing

### 4.3.1 Conversion from MSCOCO to YOLO Labels

Converting labels from MSCOCO (Microsoft Common Objects in Context) to YOLO (You Only Look Once) format is a critical step in aligning our chosen object detection dataset with the requirements of YOLO-based models. This transformation involves mapping class labels to numeric indices, converting bounding box coordinates to normalized form, and

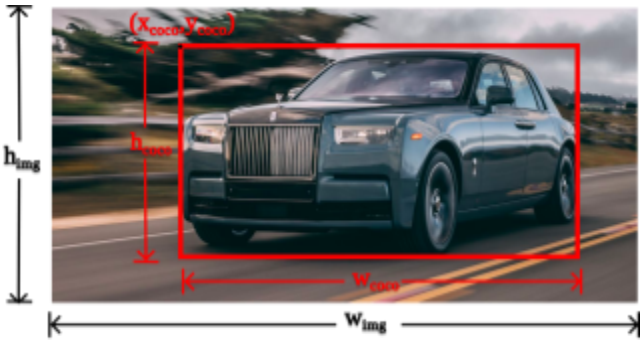



Fig 4.2 Bounding Box in COCO Format

organizing the labels in YOLO-specific text file structures. To streamline this process, custom scripts in languages like Python can be employed for automation. Verification and quality assurance checks are essential post-conversion to ensure label accuracy and adherence to YOLO's labeling standards. This conversion is pivotal in making the dataset compatible with our dynamic traffic signal management system and YOLO-based object detection framework.

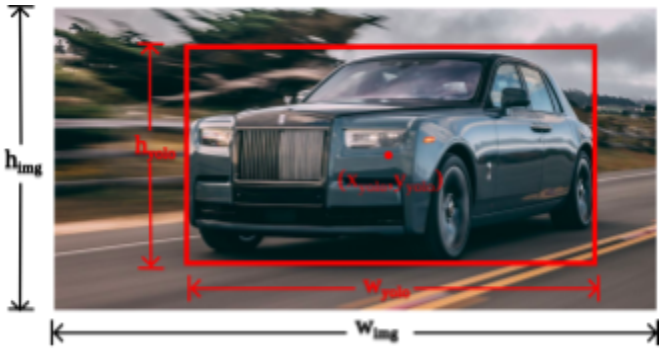



Fig 4.3. Bounding Box in YOLO Format

The following mathematical conversion was applied:

| | |
|---|---|
| $x_{yolo} = \frac{(x_{coco} + \frac{w_{coco}}{2})}{w_{img}}$ | $w_{yolo} = \frac{w_{coco}}{w_{img}}$ |
| $y_{yolo} = \frac{(y_{coco} + \frac{h_{coco}}{2})}{h_{img}}$ | $h_{yolo} = \frac{h_{coco}}{h_{img}}$ |

### 4.3.2 Splitting Dataset

The provided code snippet serves to organize and split a dataset into training and validation sets for use in machine learning applications, particularly object detection tasks like YOLO. It does so by initially defining directory structures for the original data and destination directories for the training and validation data. Subsequently, the code shuffles and splits the image files into two sets based on an 80:20 ratio, moving them to their respective sub-directories within the training and validation directories. Furthermore, it renames the sub-directories to lowercase, a common convention for label and image directories in object detection tasks. This preparatory step is crucial for structuring the

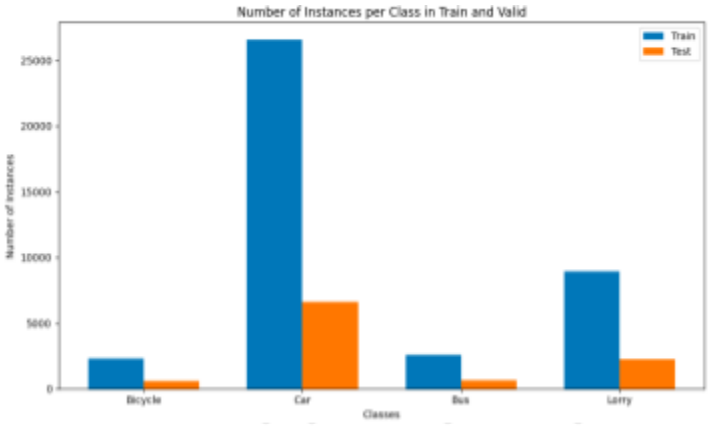



Fig 4.4. Number of Instances per Class in Train and Test

dataset in a manner compatible with YOLO and similar object detection frameworks. The number of instances per class in Train and Valid is shown in Figure 4.4.

## 4.4 Training the YOLOv8 Model

### 4.4.1 Architecture of the Model

The architecture of the model characterized is a deep neural network formulated for entity recognition, often used in applications like YOLO (You Only Look Once). This model consists of 225 layers and approximately 11.1 million parameters, making it a complex and powerful architecture. The structure includes convolutional layers, concatenation layers, and upsampling layers, which are crucial for feature extraction, multiscale detection, and object localization. The model's architecture is designed to process input images and produce bounding box predictions for objects of interest. It employs techniques such as spatial pyramid pooling (SPP) and feature concatenation to capture contextual information and enhance detection accuracy. With a capability of 28.7 GFLOPs (Giga Floating-Point Operations Per Second), this model demonstrates its efficiency in handling computationally intensive tasks like object detection. It serves as a fundamental component in the object detection pipeline, enabling precise identification and localization of objects within images.

| params | module | arguments |
|---|---|---|
| 928 | ultralytics.nn.modules.Conv | [3, 32, 3, 2] |
| 18560 | ultralytics.nn.modules.Conv | [32, 64, 3, 2] |
| 29056 | ultralytics.nn.modules.C2f | [64, 64, 1, True] |
| 73984 | ultralytics.nn.modules.Conv | [64, 128, 3, 2] |
| 197632 | ultralytics.nn.modules.C2f | [128, 128, 2, True] |
| 295424 | ultralytics.nn.modules.Conv | [128, 256, 3, 2] |
| 788480 | ultralytics.nn.modules.C2f | [256, 256, 2, True] |
| 1180672 | ultralytics.nn.modules.Conv | [256, 512, 3, 2] |
| 1838080 | ultralytics.nn.modules.C2f | [512, 512, 1, True] |
| 656896 | ultralytics.nn.modules.SPPF | [512, 512, 5] |
| 0 | torch.nn.modules.upsampling.Upsample | [None, 2, 'nearest'] |
| 0 | ultralytics.nn.modules.Concat | [1] |
| 591360 | ultralytics.nn.modules.C2f | [768, 256, 1] |
| 0 | torch.nn.modules.upsampling.Upsample | [None, 2, 'nearest'] |
| 0 | ultralytics.nn.modules.Concat | [1] |
| 148224 | ultralytics.nn.modules.C2f | [384, 128, 1] |
| 147712 | ultralytics.nn.modules.Conv | [128, 128, 3, 2] |
| 0 | ultralytics.nn.modules.Concat | [1] |
| 493056 | ultralytics.nn.modules.C2f | [384, 256, 1] |
| 590336 | ultralytics.nn.modules.Conv | [256, 256, 3, 2] |
| 0 | ultralytics.nn.modules.Concat | [1] |
| 1969152 | ultralytics.nn.modules.C2f | [768, 512, 1] |
| 2117596 | ultralytics.nn.modules.Detect | [4, [128, 256, 512]] |

Fig 4.5. Architecture of the Model

### 4.4.2 Optimizer for training

The chosen optimizer for training this model is Stochastic Gradient Descent (SGD) with a learning rate (lr) set to 0.01. This optimizer is a popular choice in deep learning and is effective for updating model weights during the training process. To manage weight decay and regularization, SGD employs specific parameter groups. Parameter Group 57 has a weight decay of 0.0, indicating no regularization for the corresponding layers. Parameter Group 64 uses a weight decay of 0.0005 and is likely associated with layers where weight regularization is essential to prevent overfitting. Parameter Group 63 is related to bias terms and applies weight decay to regularize them. This combination of parameter groups and weight decay settings aims to strike a balance between fitting the training data well and preventing overfitting, resulting in a well-generalized object detection model.

### 4.4.3 Albumentations for training

During the training process, the Albumentations library is utilized to augment and diversify the training data. These enhancements are imperative for fortifying the model's capacity to generalize effectively across diverse real-world scenarios. Several specific augmentation techniques are applied to the training data. Firstly, there is a blur augmentation (Blur) with a probability (p) of 0.01, and the blur intensity ranges between 3 and 7. Additionally, the MedianBlur augmentation (p=0.01) with similar blur intensity settings is employed. To further enhance the data, a grayscale transformation (ToGray) with a probability of 0.01 is applied, which can help the model adapt to variations in lighting conditions. Lastly, Contrast Limited Adaptive Histogram Equalization (CLAHE) is employed with a probability (p) of 0.01. This technique enhances local contrast in the images, making object features more discernible. The CLAHE parameters include the clip limit, which ranges from 1 to 4.0, and the tile grid size is set at 8x8. These augmentations collectively contribute to a more robust and adaptable object detection model during training.

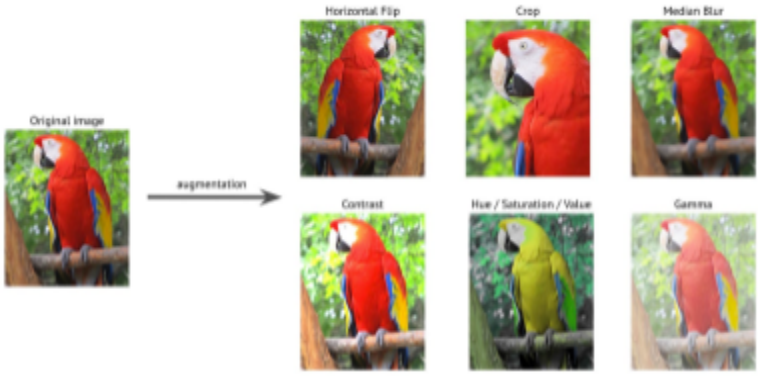



Fig 4.6. Albumentations

## 4.5 Training Evaluation and Analytics Metrics

In the domain of computer vision and YOLO (You Only Look Once) model training, performance evaluation is crucial. It involves assessing the model's proficiency in precise object detection and classification in images and videos. This section explores key evaluation metrics and analysis techniques, such as the Confusion Matrix, F1 Curve, P Curve, PR Curve, and R Curve. These indicators supply valuable discernments into the YOLO model's real-world effectiveness.

### 4.5.1 Confusion Matrix

The Confusion Matrix, in this context, is a 5x5 matrix that helps assess the effectiveness of YOLO models in discerning objects' possession to the classes 'Bicycle,' 'Car,' 'Bus,' 'Lorry,' and the 'Background.' It offers an all-encompassing perspective on the extent of conformity between the model's predictions and the authentic data for each category.

Fig 4.7. Confusion Matrix

### 4.5.2 F1-Confidence Curve

The F1-Confidence Relationship Graph visually depicts how adjustments in the confidence threshold impact the F1-score within the context of object detection in YOLO models. The F1-score is a measure of the model's accuracy in simultaneously evaluating both precision and recall, making it a valuable metric for assessing object detection performance. By varying the confidence threshold, which determines when a detection is considered valid, this curve provides insights into how the model's precision and recall trade off against each other at different confidence levels. It helps users identify an optimal threshold that balances the need for high precision and high recall, depending

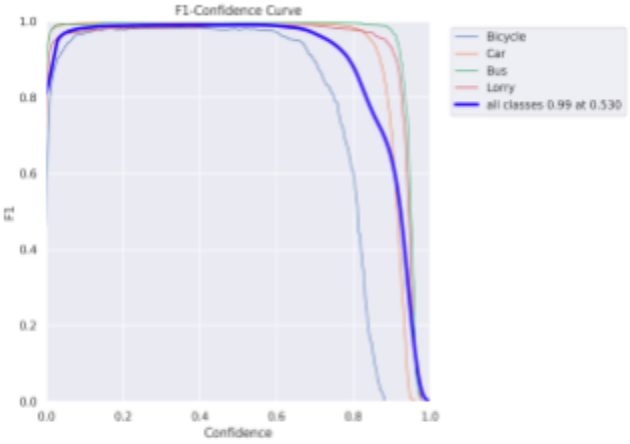



Fig 4.8. F1-Confidence Curve

on their specific application requirements. This curve is a valuable tool for fine-tuning YOLO models to achieve the desired balance between accuracy and robustness in object detection.

### 4.5.3 Precision-Confidence Curve

The Precision-Confidence Curve is a vital visualization tool for evaluating the precision of a YOLO model's predictions across varied confidence levels. This graphical representation aids in comprehending the fluctuation in the model's accuracy as the confidence threshold for object detection undergoes adjustment. By plotting precision against confidence scores, we can identify the threshold values that yield the desired level of precision for specific tasks or applications. This curve is particularly valuable for making informed decisions about trade-offs between precision and recall, allowing users to fine-tune the model's performance based on their specific requirements.


Fig 4.9. Precision-Confidence Curve

### 4.5.4 Precision-Recall Curve

The Precision-Recall Curve is a vital evaluation tool in assessing the YOLO model's effectiveness in object localization. This graphical representation illustrates the balance between accuracy and inclusivity at varying confidence thresholds, providing a visual depiction of the model's capacity to accurately identify objects while maximizing the retrieval of pertinent instances. Precision represents the percentage of correctly identified positive detections within the set of positive forecasts, emphasizing the model's accuracy. Recall, on the other hand, quantifies the model's ability to detect all confirmed positive cases within the dataset, focusing on its completeness. By analyzing this curve, we gain insights into how the model's confidence threshold impacts its

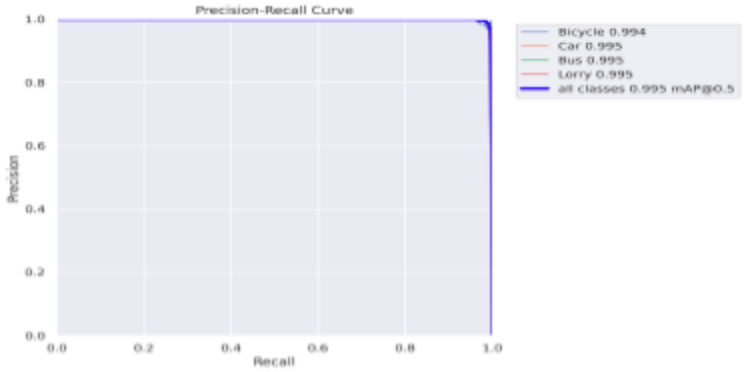



Fig 4.10. Precision-Recall Curve

precision and recall rates, allowing us to make informed decisions about the ideal threshold for our specific application.

### 4.5.5 Recall-Confidence Curve

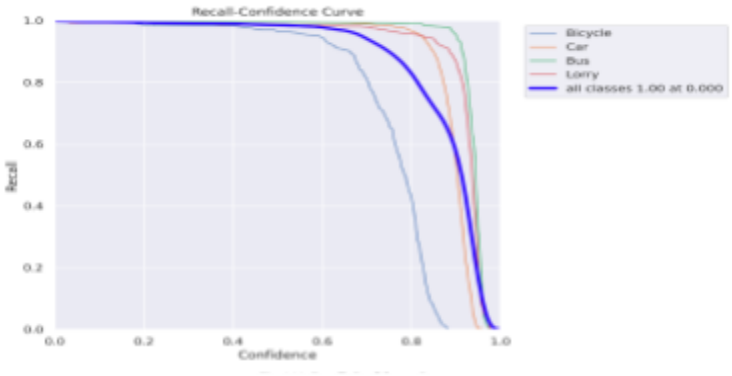



Fig 4.11. Recall-Confidence Curve

The Recall-Confidence Curve is a crucial visualization that showcases the recall of the YOLO model across different levels of confidence thresholds. In this curve, the x-axis illustrates the varying confidence thresholds, while the y-axis illustrates the corresponding recall values. The recall quantifies the ability of the model to correctly detect all instances of a specific class within the dataset. By analyzing this curve, we can gain insights into how different confidence thresholds impact the model's recall performance for each object class. This information is invaluable for fine-tuning the model's detection capabilities and optimizing its performance for specific classes or applications.

### 4.5.6 Training result

Metrics play a key function in assessing the effectiveness of the YOLO model in object detection tasks, aiding in both training and evaluation phases:

- **Train/Box_Loss:** The train/box_loss is a critical metric during the training phase of the YOLO model. It measures the loss associated with bounding box predictions. Lower values indicate that the model is effectively learning to localize objects in the images.
- **Train/Cls_Loss:** Train/cls_loss quantifies the loss related to class predictions during model training. It signifies how well the model is learning to classify objects. A decreasing cls_loss indicates improved class prediction accuracy.
- **Train/Dfl_Loss:** The train/dfl_loss measures the loss associated with depth-wise focal loss during training. This is especially relevant for object detection tasks where certain classes may be challenging to distinguish, and it helps the model focus on difficult cases.
- **Metrics/Precision(B):** Precision(B) is a crucial metric calculated during evaluation. It signifies the precision of bounding box predictions. High precision values indicate that a majority of the predicted bounding boxes are accurate.

- **Metrics/Recall(B):** Recall(B) is another evaluation metric. It measures the ability of the model to recall or detect all relevant objects in the dataset. A high recall value suggests that the model is successful at recording most instances of objects.
- **Val/Box_Loss:** Similar to train/box_loss, val/box_loss measures the bounding box prediction loss during the validation phase. It helps assess how well the model generalizes to unseen data.
- **Val/Cls_Loss:** Val/cls_loss quantifies the class prediction loss on the validation dataset. It provides insights into the model's ability to classify objects accurately on unseen data.
- **Val/Dfl_Loss:** Val/dfl_loss is the depth-wise focal loss computed during validation. It ensures that the model maintains focus on challenging cases when making predictions on new data.
- **Metrics/mAP50(B):** Metrics/mAP50(B) stands for "mean Average Precision at IoU threshold of 0.5" for bounding boxes. It calculates the average precision for different object classes, considering predictions with an Intersection over Union (IoU) of 0.5 or higher as correct detections.
- **Metrics/mAP50-95(B):** Metrics/mAP50-95(B) represents the mean Average Precision over a range of IoU thresholds from 0.5 to 0.95. It provides a comprehensive evaluation of the model's performance, considering various IoU thresholds.

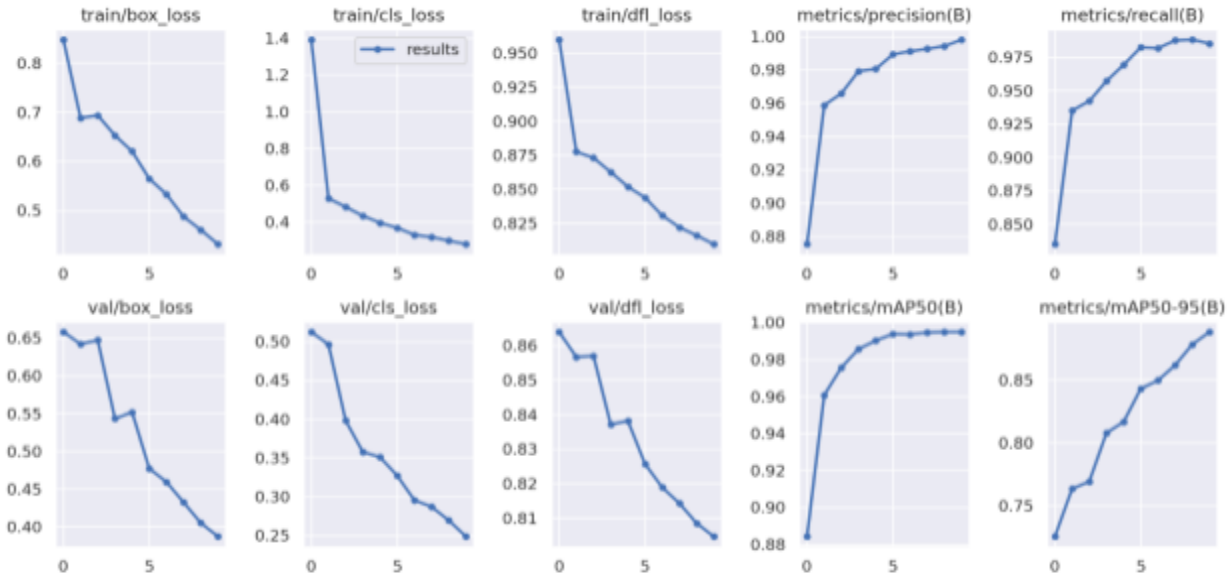



Fig 4.12. Training Result

## 4.6 Average Vehicle Duration

Traffic monitoring systems rely on the calculation of the average duration a vehicle takes to traverse a specific intersection, a metric crucial for optimizing traffic flow, predicting congestion, and enhancing road safety. This process involves a multi-step methodology starting with the use of YOLO, a powerful object detection algorithm. YOLO facilitates real-time identification and tracking of vehicles, providing precise location data within the video frames.

To narrow the focus to relevant areas of the traffic scenario, Regions of Interest (ROIs) are marked on connecting roads. These ROIs delineate specific sections where vehicle movements are of particular interest. The Point In Polygon algorithm is then applied to monitor vehicle entries and exits from these defined ROIs. This algorithm efficiently determines whether a given point, representing the vehicle's position, is inside the designated polygon.

The crux of the analysis lies in calculating the Average Vehicle Duration, achieved by dividing the number of frames a vehicle spends inside the ROI by the frame rate. This computation yields the time duration it takes for an automobile to travel across the marked region. The formula is expressed as:

$$Average\ Vehicle\ Duration = \frac{Number\ of\ Frames\ Inside\ ROI}{Frame\ Rate}$$

To provide a more nuanced perspective, vehicles are classified into different types (e.g., cars, trucks, motorcycles), and the Average Vehicle Duration is computed separately for each class. This classification allows for a more detailed analysis of traffic dynamics. Understanding the average duration for distinct vehicle classes is pivotal for implementing effective traffic management strategies. The information derived from this analysis can inform signal timing adjustments, assess the impact of traffic interventions, and contribute to predictive models for traffic patterns.

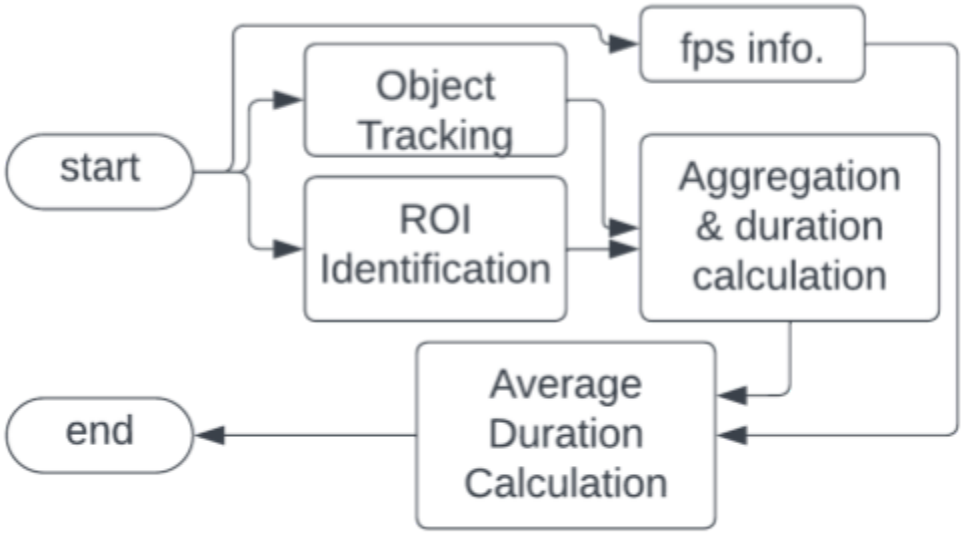



Fig 4.13. Workflow of average speed calculation

### 4.6.1 Algorithm

The algorithm for calculating the average vehicle duration to cross the intersection for each vehicle class as follows:

1. Initialize necessary components and dependencies.
2. Load pre-trained object detection model.
3. Open video source for processing.
4. Create an empty data structure to store tracking information.
5. Loop through each frame in the video:
    a. Use object detection model for tracking.
    b. Retrieve relevant tracking data.
    c. Store the tracking data.
6. Prepare and structure the tracking data.
7. Define the area of interest in the video.
8. Implement a method to determine if a point is inside the area of interest.
9. Iterate through tracking data:
    a. Identify when objects enter and exit the area of interest.
    b. Calculate the duration of each object's presence in the area.
10. Filter out irrelevant data and organize by object class.
11. Convert durations to a common time unit.
12. Print or visualize the average duration for each object class.
13. Release resources and cleanup.

## 4.7 Simulation

In tandem with the analytical insights gained from the YOLO model's vehicle tracking and duration calculations, a dynamic and interactive dimension is introduced through the implementation of a simulation using Pygame. This simulation serves as a visual representation of the proposed Adaptive Traffic Signal Management strategy. Leveraging the real-time data collected by the YOLO model, the simulation employs adaptive signal control mechanisms to respond intelligently to varying traffic conditions at the intersection. Through Pygame's interactive and graphical capabilities, the simulated environment vividly demonstrates the fluid adjustment of signal timings, optimizing traffic flow, and mitigating congestion. This dynamic approach not only enhances the comprehensibility of the proposed strategy but also provides a tangible representation of its potential impact on real-world traffic scenarios. The combination of cutting-edge computer vision analysis and interactive simulation constitutes a robust methodology for exploring and validating adaptive traffic signal management strategies.

### 4.7.1 Key steps in the development of the Simulation

Our traffic simulation project involved a meticulous process that seamlessly integrated various elements to generate a realistic and fluctuating setting. The development journey comprised the following key steps:

1. **Background Image Integration:**
    - Utilizing an image of a crossroads as the foundational environmental setting for the simulation.
2. **Vehicle and Traffic Signal Imagery:**
    - Curating top-view images of diverse vehicle types, resizing and rotating them for accurate representation.
    - Collecting images for traffic signals in red, yellow, and green states.
3. **Rendering Signals and Timing Management:**
    - Implementing code for rendering traffic signals based on their current state.
    - Dynamically managing signal timing, factoring in the algorithm and the number of vehicles present.
4. **Dynamic Vehicle Generation:**
    - Generating vehicles with randomized attributes such as direction, lane, and turning preferences.
    - Controlling the distribution of vehicles among the four directions for a balanced simulation.

5. **Intelligent Vehicle Movement:**
    - Coding the movement behavior of vehicles with varying speeds, considering gaps between vehicles, and adjusting speeds based on the presence of larger vehicles.
6. **Traffic Signal Rules Implementation:**
    - Defining how vehicles react to traffic signals, including stopping for red, moving for green, and responding to yellow signals.
    - Managing scenarios where vehicles have already crossed the stop line when the signal changes.
7. **Informative Display Features:**
    - Implementing features to display the count of automobiles that have crossed the signal.
    - Presenting the time passed since simulation initiation for enhanced monitoring.
8. **Simulation Progression and Exit Conditions:**
    - Developing code for updating elapsed time as the simulation progresses.
    - Establishing conditions for simulation termination when the elapsed time matches the desired simulation duration.
9. **Lane Addition for Realism:**
    - Enhancing realism by incorporating an additional lane to the left, dedicated to bikes only.
10. **Turning Vehicles for Realistic Dynamics:**
    - Implementing logic for vehicles to make turns and cross the intersection, adding a layer of authenticity to the simulation.

### 4.7.2 Pseudocode/Algorithm

The pseudocode/algorithm to simulate Adaptive Traffic Light System at a traffic intersection as follows:

1. **Simulation Configuration:**
    - Initialize default values for signal timings, vehicle times, and simulation parameters
    - Create data structures to store information about traffic signals, vehicles, and simulation state
2. **Traffic Signal Class:**
    - Initialize signal attributes (red, yellow, green, minimum, maximum, totalGreenTime)
    - Define methods to manipulate and query signal attributes
3. **Vehicle Class:**

- Initialize vehicle attributes (lane, vehicleClass, speed, direction, etc.)
- Define methods for vehicle movement, turning, and initialization

4. **Initialization:**
   - Initialize traffic signals with default values
   - Start the main simulation loop
5. **Set Time:**
   - Determine the time for the next green signal based on the count and types of waiting automobile
6. **Repeat:**

   Loop to simulate traffic:
   - Update signal timings
   - Display current status
   - Move vehicles based on traffic rules and signal states
   - Check for turning vehicles and apply turning logic
   - If it's time to transition to the next signal, set the next signal as green

   Repeat the simulation loop
7. **Print Status:**
   - Display the status of each traffic signal, including time remaining for red, yellow, and green signals
8. **Update Values:**
   - Decrement the time for each signal state and update the total green time
9. **Generate Vehicles:**
   - Generate random vehicles with various types, lanes, and turning preferences
10. **Simulation Time:**
    - Monitor overall simulation time and print statistics when the simulation ends
11. **Main:**
    - Initialize the Pygame simulation environment
    - Start threads for initialization, vehicle generation, and simulation time monitoring
    - Display the background and signal images
    - Render text for signal timers and vehicle counts
    - Animate the movement of vehicles

### 4.7.3 Simulation Visualizations

In this section, we present visualizations of the dynamic traffic signal duration management system implemented using YOLOv8. The screenshots below depict the simulation environment and the adaptive traffic control mechanisms in action.

#### 4.7.3.1 Simulation Environment Setup

The setup of the simulation environment is meticulously crafted to mirror real-world traffic dynamics. Central to this setup is the configuration of traffic signals, where default timings for red, yellow, and green signals are established. These timings orchestrate the flow of vehicles through intersections, dictating when each lane is granted right of way. Alongside signal settings, parameters defining vehicle behavior are crucially defined, encompassing average speeds for distinct vehicle types such as cars, buses, and bikes. These speeds are pivotal in simulating vehicle movements accurately within the environment, influencing lane navigation and intersection traversal.

Furthermore, the simulation environment setup involves precise delineation of spatial coordinates for lanes, intersections, and stop lines. These coordinates serve as the framework for positioning vehicles and traffic signals, ensuring an accurate emulation of real-life traffic situations. Complemented by visualization components using Pygame, the environment offers a graphical representation of traffic dynamics, including signal states and vehicle movements. With control parameters meticulously defined, such as simulation time and gap distances between vehicles, the environment as shown in Fig x.x is primed for comprehensive modeling and analysis, offering insights into effective traffic signal management strategies and their impact on overall traffic flow.

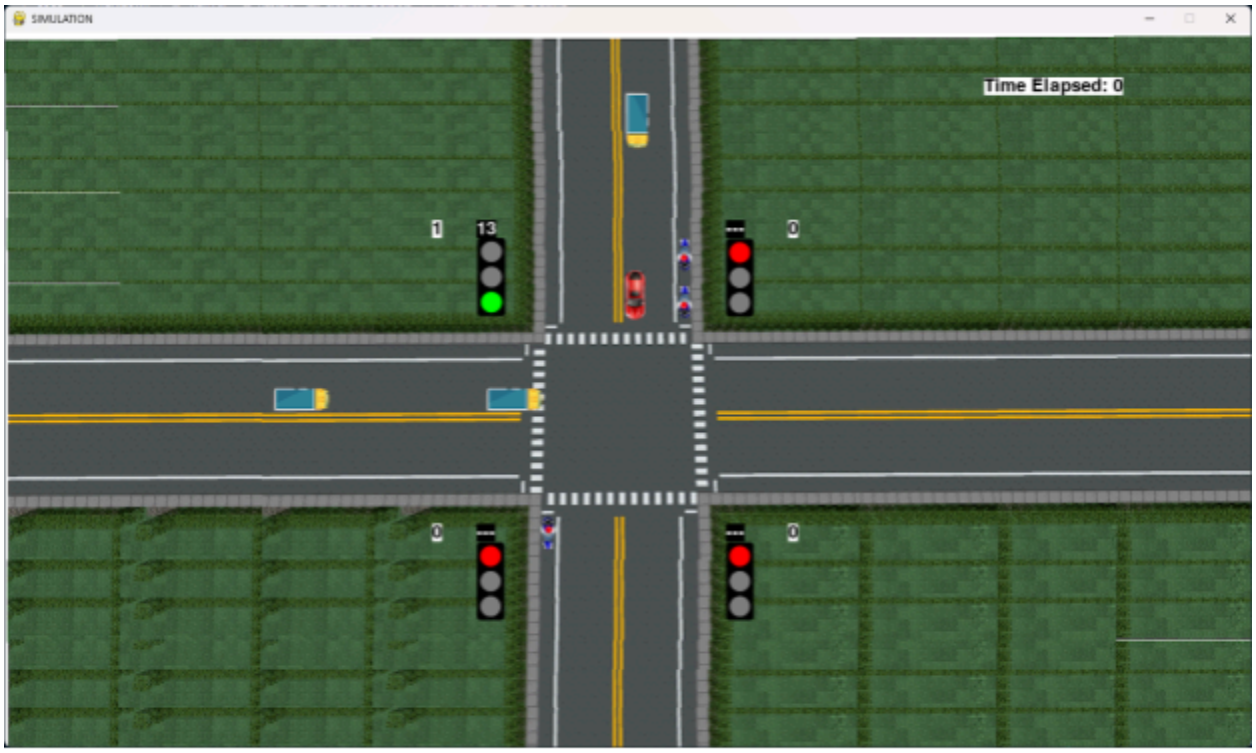



Fig 4.14. Simulation Environment Setup

#### 4.7.3.2 Dynamic Traffic Signal Control

The implementation of dynamic traffic signal control leverages an adaptive approach to regulate traffic flow effectively within the simulated environment. At the core of this methodology lies the real-time adjustment of signal timings based on prevailing traffic conditions. Through vehicle detection mechanisms integrated into the simulation, the system continuously monitors lane occupancy and vehicle densities, crucial indicators for signal adjustment. Upon detection of varying traffic volumes at different intersections, the system dynamically recalibrates signal timings to optimize traffic throughput.

Utilizing a centralized control mechanism, the system evaluates traffic data collected from each lane and intersection, orchestrating signal transitions accordingly. The decision-making process involves a sophisticated algorithm that analyzes the distribution and density of vehicles across lanes, determining optimal green signal durations for each intersection. By dynamically adapting signal timings in response to changing traffic patterns, the system mitigates congestion, minimizes wait times, and enhances overall traffic efficiency. This adaptive control strategy ensures seamless traffic flow through intersections, effectively addressing fluctuations in demand and maximizing the utilization of road networks. Through iterative refinement and optimization, the dynamic traffic signal control system demonstrates its efficacy in managing traffic dynamics and improving the overall commuting experience.

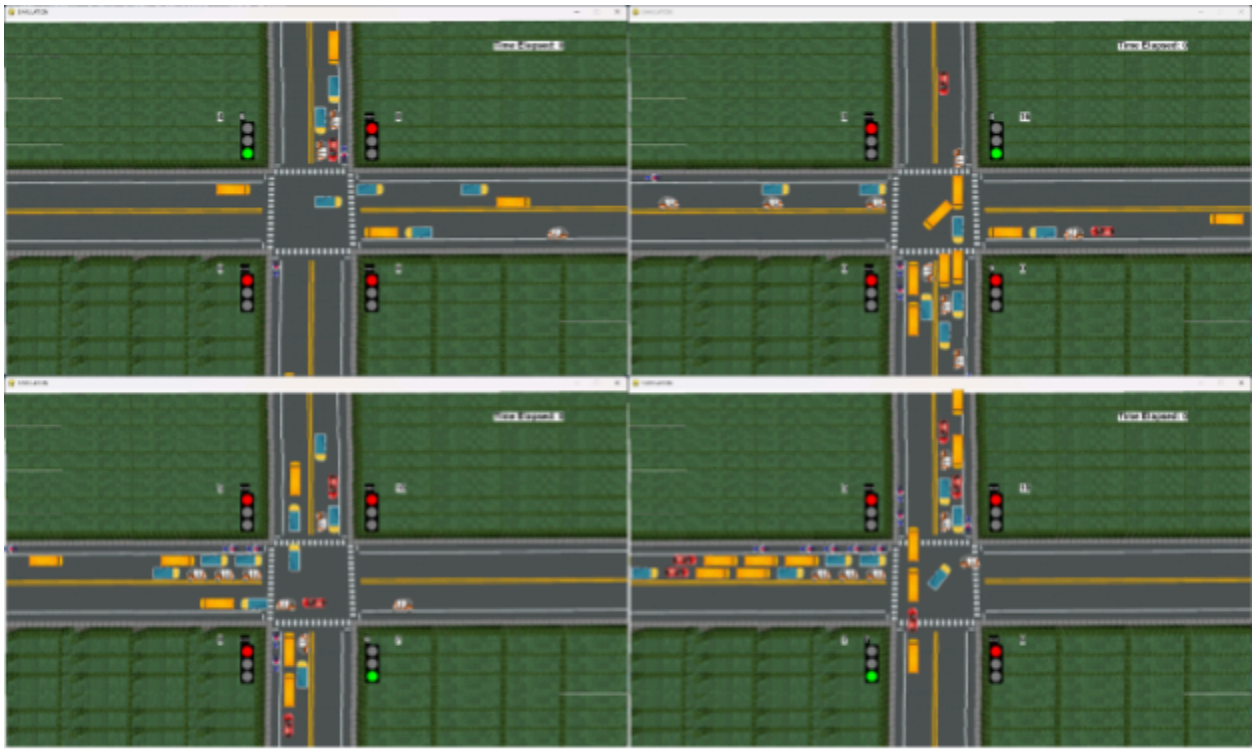


Fig 4.15. Dynamic Traffic Signal Control

# CHAPTER 5
# RESULT AND DISCUSSION

In this section, we delve into the outcomes of our comprehensive experiments and analyses conducted with the YOLO (You Only Look Once) entity identification model. The preceding sections have detailed the architecture, training methodologies, and evaluation metrics. Now, it is time to unveil the fruits of our labor and engage in insightful discussions. We present the model's computational effectiveness on diverse entity identification tasks, showcasing its ability to precisely localize & classify objects within images. Furthermore, we explore the implications of the obtained results, considering the challenges and opportunities presented by the YOLO model. Through a detailed examination of the results and discussions, we aim to provide valuable insights into the capabilities and limitations of YOLO for real-world object detection applications.

## 5.1 Model Performance Summary

After training and evaluating the YOLO object detection model on our dataset, we have obtained a comprehensive performance summary as shown in Figure 5.1. The model's performance is assessed across various classes, including "Bicycle," "Car," "Bus," "Lorry," and the "Background" class. The summary includes key metrics such as precision (P), recall (R), Average Precision calculated at IoU threshold 0.5 (mAP50), and Average Precision computed across IoU values from 0.5 to 0.95 (mAP50-95). These metrics provide valuable insights into the classification precision for diverse object recognition. It is evident that the model achieves high precision and recall rates across all classes, indicating its effectiveness in entity identification. The mAP50 and mAP50-95 scores further highlight the model's resilience and accuracy in handling a wide range of objects in real-world scenarios. This performance summary serves as a critical evaluation of the YOLO model's capabilities in our specific application.

| Class | Images | Instances | Box(P | R | mAP50 | mAP50-95): 100% 22/22 [00:31<00:00, 1.44s/it] |
|---|---|---|---|---|---|---|
| all | 347 | 5594 | 0.99 | 0.978 | 0.992 | 0.862 |
| Bicycle | 347 | 410 | 0.969 | 0.949 | 0.985 | 0.578 |
| Car | 347 | 3632 | 0.997 | 0.989 | 0.995 | 0.936 |
| Bus | 347 | 323 | 0.998 | 0.994 | 0.995 | 0.987 |
| Lorry | 347 | 1229 | 0.995 | 0.982 | 0.995 | 0.946 |

Fig 5.1. Model Performance

## 5.2 Testing Evaluation and Evaluation Metrics

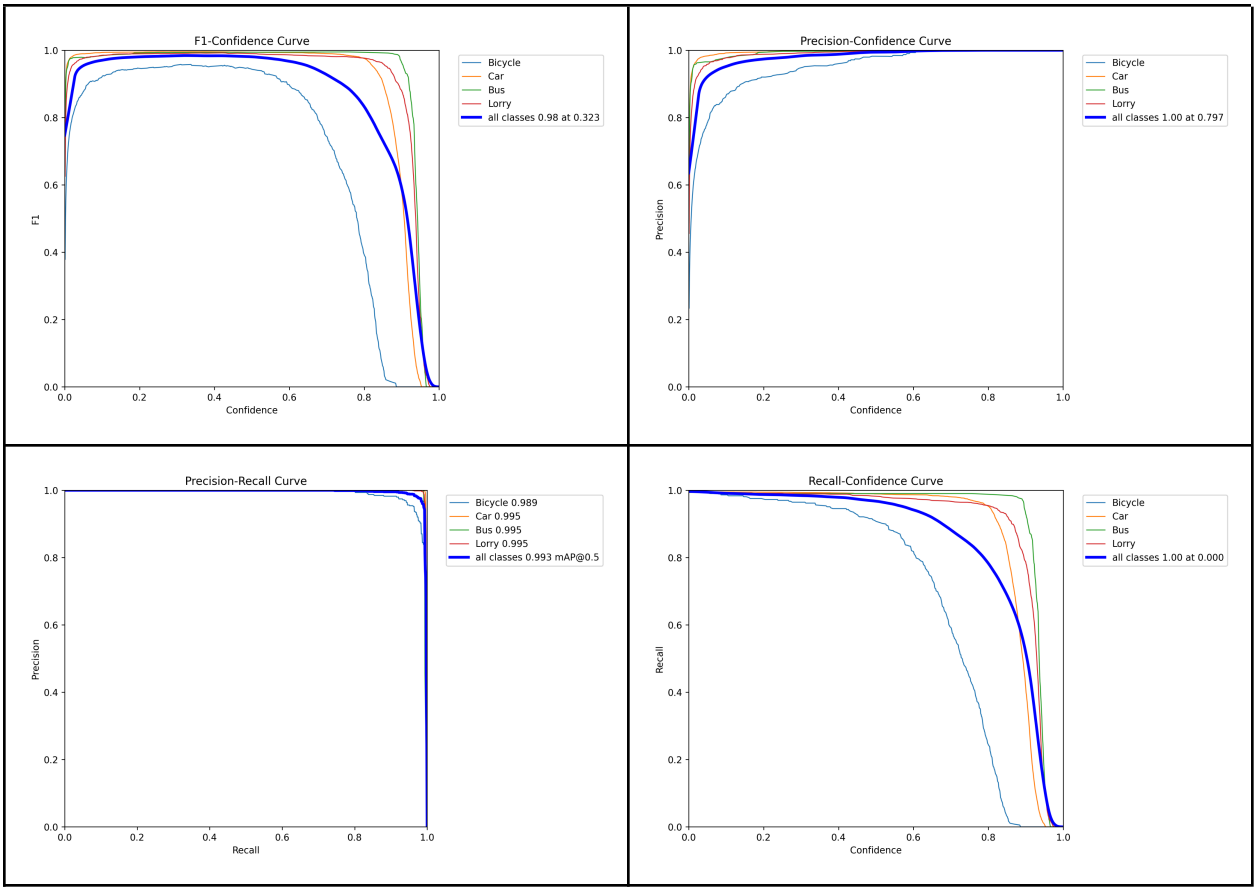



Fig 5.2. Testing Result

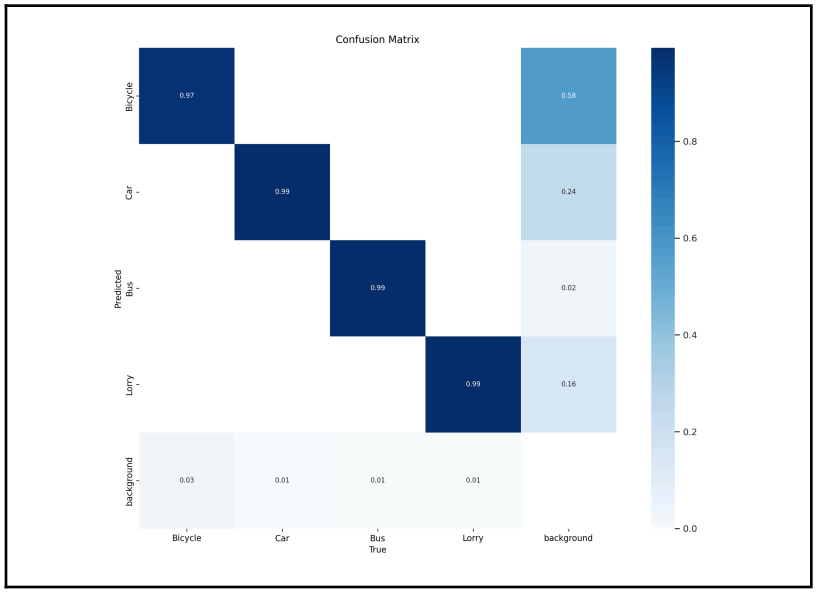



Fig 5.3. Testing Confusion Matrix

## 5.3 Model Output

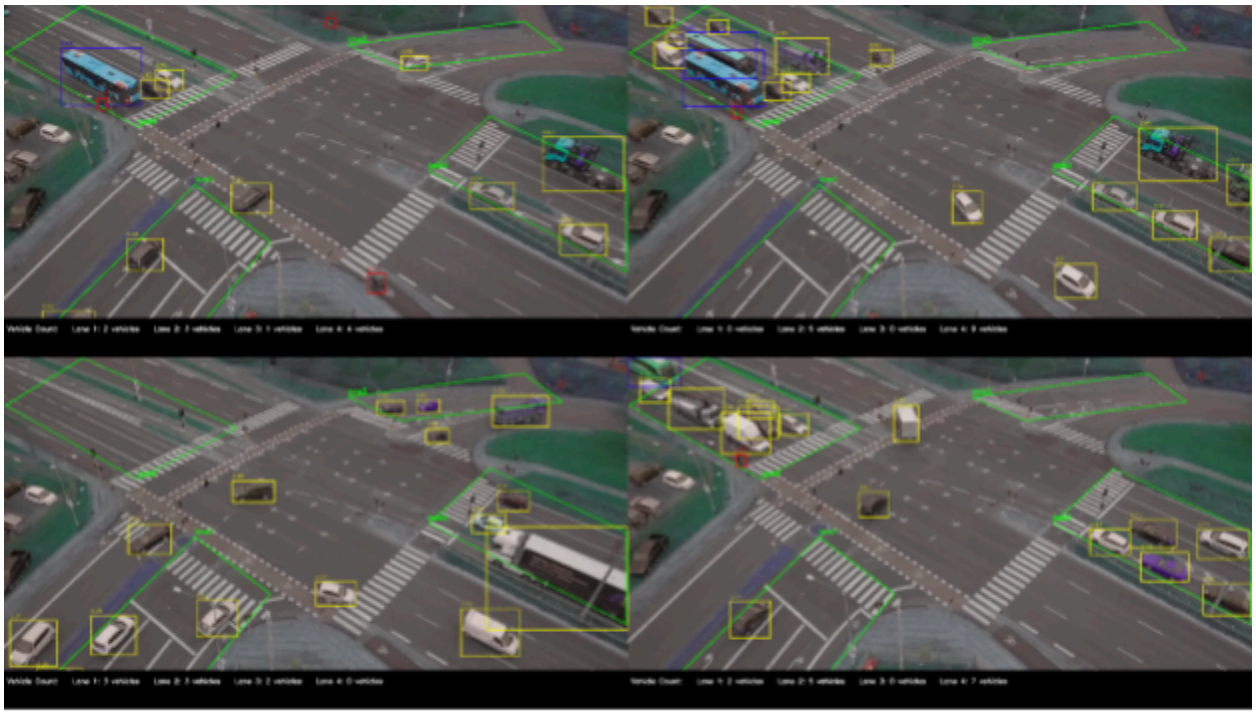


Fig 5.4. Model Output with Vehicle Count per Lane

## 5.4 Average Duration of each Vehicle Class

In the realm of traffic monitoring, the average duration of vehicles passing through a designated intersection offers valuable insights into traffic dynamics. Utilizing the YOLO object detection model, this algorithm precisely tracks and analyzes the movement of different vehicle classes within a specified Region of Interest (ROI). The resulting average durations provide a concise measure of traffic efficiency and congestion, enabling informed decisions for urban mobility enhancement.

The following results were obtained:

TABLE 5.1. AVERAGE DURATION OF EACH VEHICLE CLASS

| Class | Average Frame Duration (frames) | Average Duration (secs) |
|---|---|---|
| **Bicycle** | 106.000000 | 3.533333 |
| **Car** | 96.309524 | 3.210317 |
| **Bus** | 102.000000 | 3.400000 |
| **Lorry** | 113.750000 | 3.791667 |

## 5.5 Simulation Results and Analysis

The simulation results provide a comparative analysis between a Basic Traffic Intersection with fixed signal timings and an Adaptive Traffic Intersection employing dynamic signal control strategies. In the case of the Basic Traffic Intersection, where each road segment is allocated a static 10-second Green Time, the average traffic flow rates for individual road segments and the entire intersection are assessed.

### 5.5.1 Traffic Flow Analysis Results

Traffic flow analysis is a fundamental practice within transportation engineering, focusing on the systematic evaluation of vehicle movement within road networks and intersections. Traffic flow analysis involves the systematic evaluation of vehicle movement within road networks and intersections, focusing on parameters like speed, density, and flow rates. Its significance lies in informing decisions related to infrastructure design, signal timings, and resource allocation, ultimately contributing to more efficient transportation systems. By identifying congestion points and optimizing operations, it plays a crucial role in reducing travel times and enhancing overall urban mobility.

The simulation results illustrate the contrasting performance of basic and adaptive traffic intersection setups. In the basic scenario with fixed signal timings, traffic flow rates varied across roads. However, the adaptive system, dynamically adjusting signals, demonstrated improved traffic flow as shown below:

TABLE 5.2. TRAFFIC FLOW ANALYSIS

| Class | Basic Traffic Intersection (vehicle/sec) | Adaptive Traffic Intersection (vehicle/sec) |
|---|---|---|
| **Road 1** | 1.3558 | 1.4918 |
| **Road 2** | 1.40384 | 1.4413 |
| **Road 3** | 0.9659 | 1.5 |
| **Road 4** | 0.9846 | 1.5221 |
| **Whole Intersection** | 1.1775 | 1.4888 |

To visually analyze the traffic flow results, we'll create a simple bar chart. The chart will compare average traffic flow rates for each road and the intersection in both basic and adaptive traffic setups. Roads as well as the whole intersection will be on the x-axis, and flow rates (vehicles per second) on the y-axis.

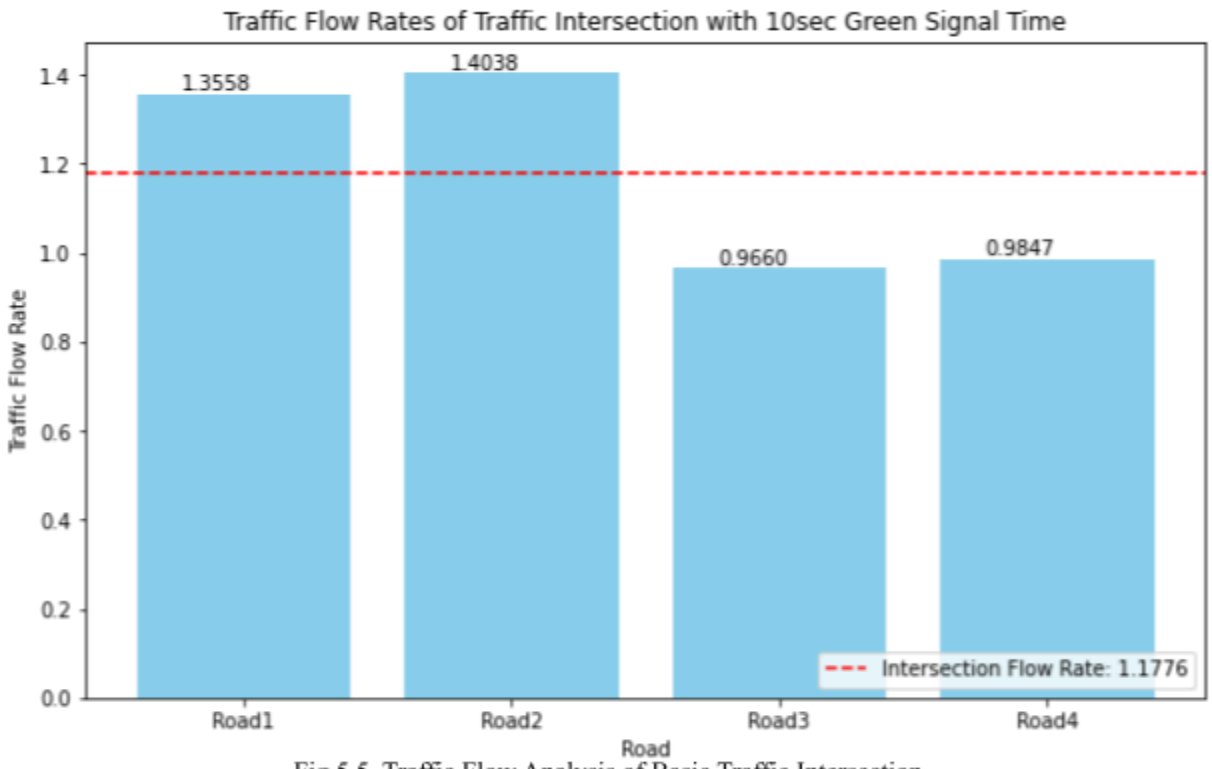



Fig 5.5. Traffic Flow Analysis of Basic Traffic Intersection

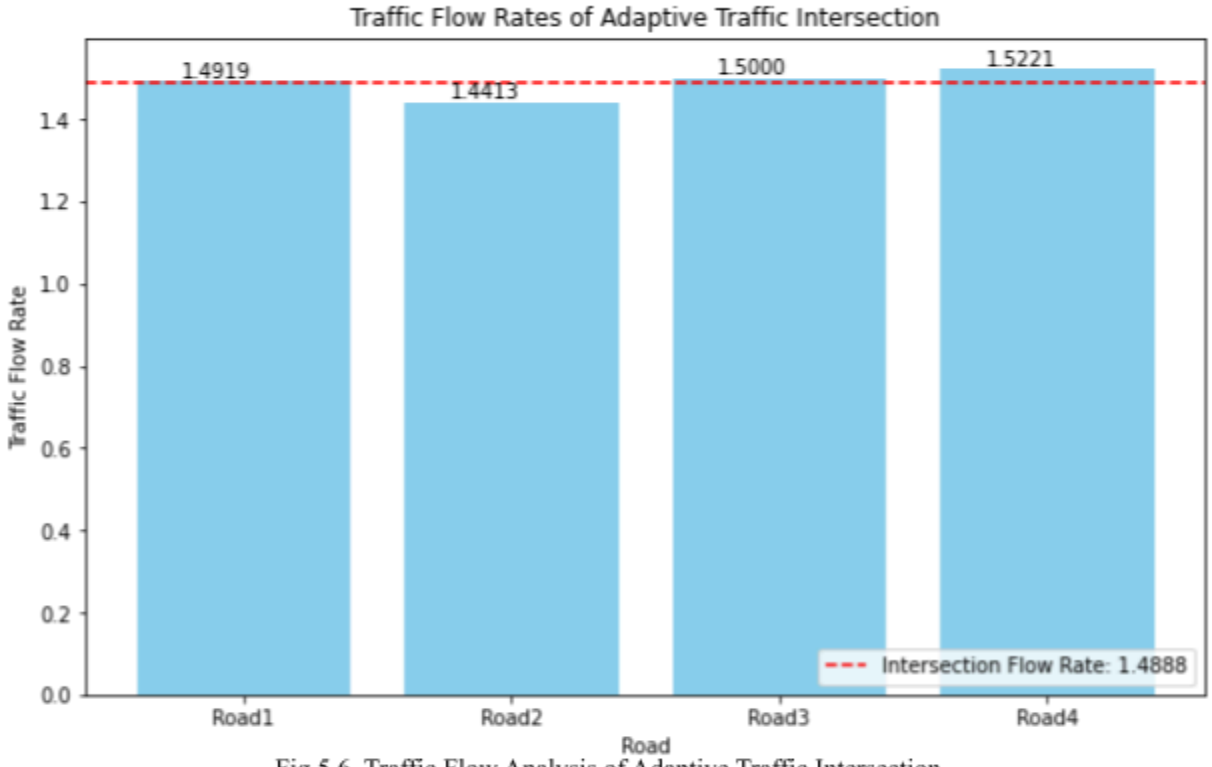



Fig 5.6. Traffic Flow Analysis of Adaptive Traffic Intersection

### 5.5.2 Box Plot Analysis Results

The box plot analysis offers valuable insights into the performance of the adaptive traffic intersection. Each box in the plot depicts the distribution of the count of cars exited during the green light period for a specific road. By visually comparing these distributions, it becomes possible to identify any variations in traffic flow patterns across different roads within the intersection. Additionally, annotations for minimum, maximum, and average values provide a comprehensive overview of the traffic flow characteristics for each road. Horizontal lines indicating the minimum, maximum, and average values for the entire intersection enable a comparison between individual road performance and the overall intersection performance.

The following is the Box plot comparison between both the traffic intersection models:

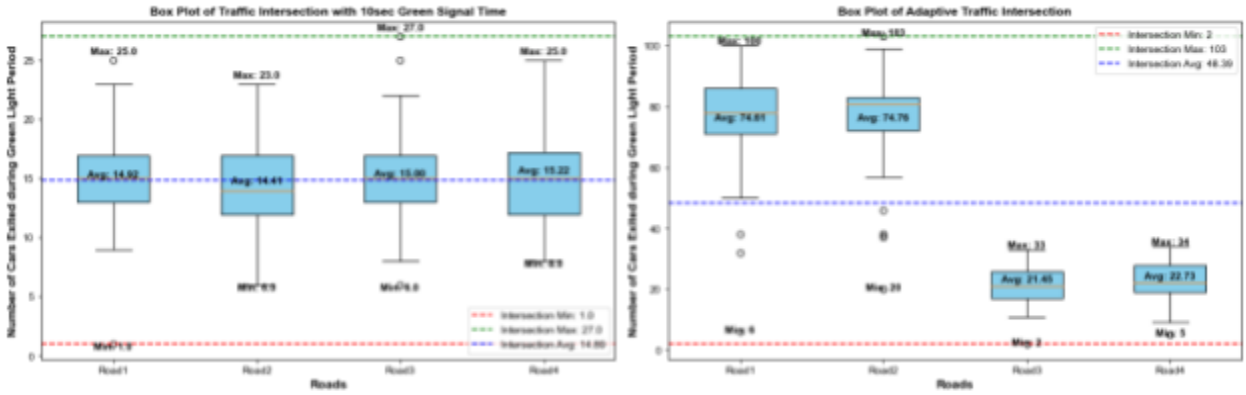



Fig 5.7. Box Plot Analysis of both the Traffic Intersection Models

### 5.5.3 Distribution Analysis Results

The distribution analysis results provide further insights into the traffic flow dynamics within the intersection. By examining the distribution of the number of cars exited during the green light period for each road, it becomes possible to discern the shape, spread, and central tendency of the traffic flow distribution. This analysis helps in understanding the variability and skewness of the traffic flow data, which is crucial for devising effective traffic signal control strategies. Additionally, comparing the distribution of traffic flow across different roads enables the identification of any asymmetry or outliers in the data, which may indicate potential bottlenecks or congestion points.

The Traffic Distribution of Basic Traffic Intersection as follows:

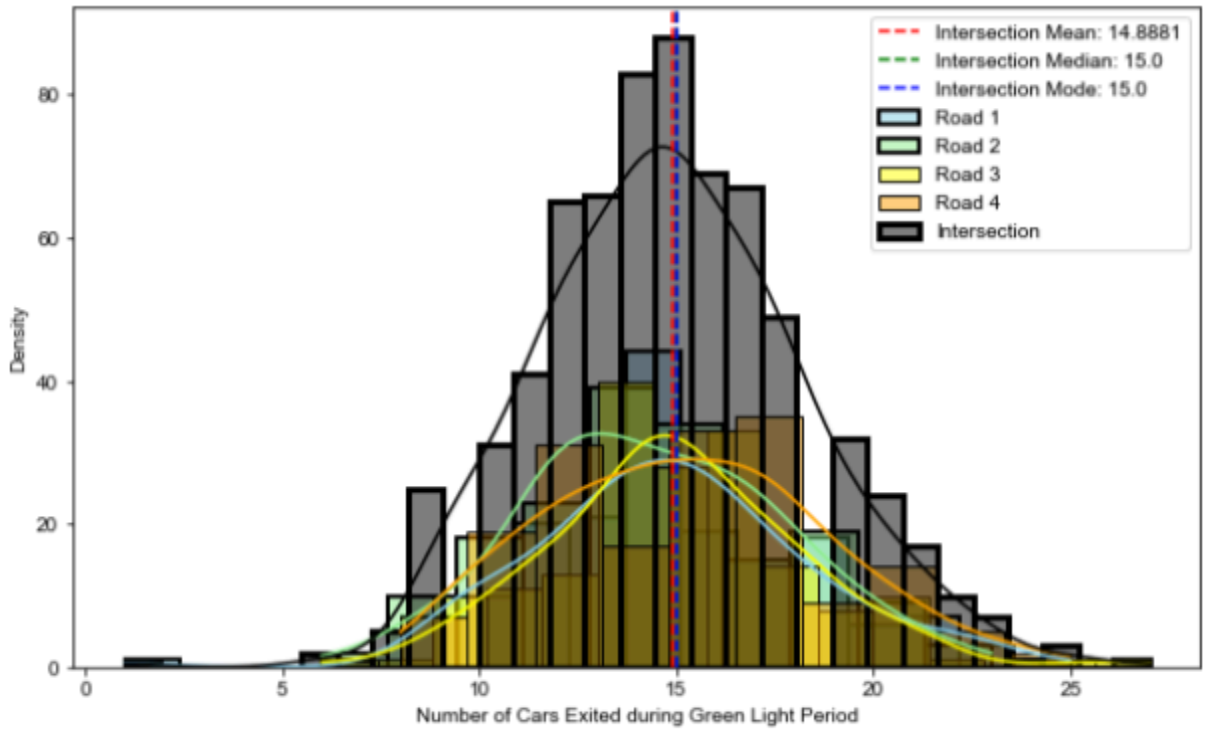



Fig 5.8. Skewed Distribution of Basic Traffic Intersection

The Traffic Distribution of Adaptive Traffic Intersection as follows:

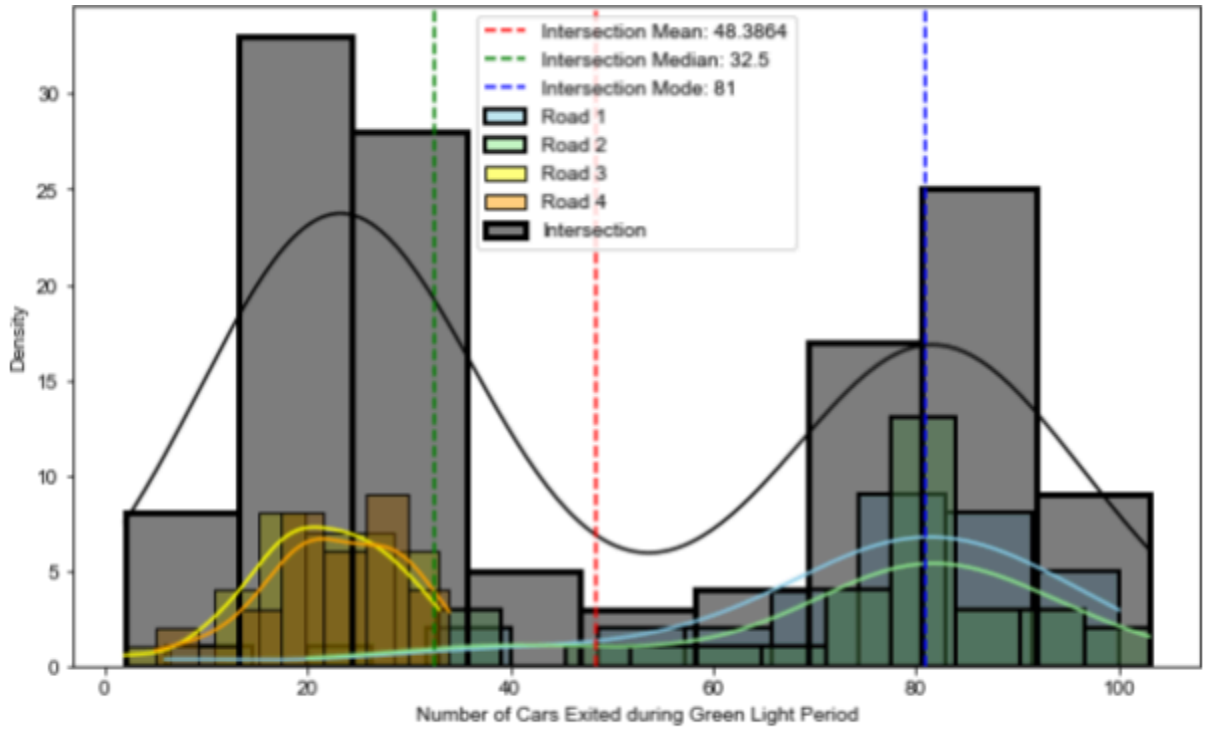



Fig 5.9. Skewed Distribution of Adaptive Traffic Intersection

The occurrence of two peaks in the skewed distribution of traffic flow at the Adaptive Traffic Intersection reflects a dynamic response to the nuanced patterns of traffic demand. This adaptability allows the intersection to efficiently manage not only the typical rush hours but also secondary peaks that may arise due to factors such as staggered work schedules, school dismissals, or localized events. By accommodating these secondary peaks, the adaptive system demonstrates its robustness in addressing diverse traffic scenarios, ultimately leading to smoother traffic flow and reduced congestion throughout the day.

Moreover, the presence of two peaks signifies improved utilization of roadway capacity and resources. By optimizing signal timings to align with varying traffic patterns, the adaptive traffic control system maximizes the efficiency of the intersection, ensuring that available road space is effectively utilized during both primary and secondary peak periods. This optimization not only enhances the throughput of vehicles but also minimizes delays and queuing, resulting in more predictable travel times for commuters and a more sustainable use of transportation infrastructure.

# CHAPTER 6
# CONCLUSION AND SCOPE OF FUTURE WORK

## 6.1 Conclusion

In this study, we presented a comprehensive analysis of the YOLO (You Only Look Once) object detection model's performance for real-time vehicle detection in traffic surveillance scenarios. The model was trained and evaluated on a rich and diverse dataset, capturing various vehicle types, lighting conditions, and traffic scenarios. The experimental findings showcased the efficacy of the YOLO model in accurately detecting and classifying vehicles, including bicycles, cars, buses, and lorries. The model achieved high precision and recall rates across all classes, showcasing its robustness in handling complex urban traffic environments.

Furthermore, the analysis of metrics such as mean Average Precision (mAP) at different IoU thresholds and the Confusion Matrix provided valuable perceptions into the model's behavior and its ability to handle multi-class object detection tasks. The YOLO model's high performance and efficiency make it a promising candidate for various applications in urban traffic management, intelligent transportation systems, and adaptive traffic signal control.

In conclusion, our research highlights the potential of deep learning-based object detection techniques, particularly YOLO, in enhancing urban traffic monitoring and management. Future work could focus on further fine-tuning the model and integrating it into real-world traffic control systems to achieve safer and more efficient urban mobility.

## 6.2 Scope for future work

While this study has provided valuable insights into the application of the YOLO object detection model in traffic surveillance and management, there exist numerous prospects for additional investigation and advancement within this field:

- **Real-Time Deployment:** One of the immediate areas for further work is the real-time deployment of the YOLO model in traffic signal control systems. Integrating the model into existing traffic management infrastructure and assessing its real-time performance under dynamic traffic conditions would be a crucial step.
- **Traffic Flow Optimization:** Future research could focus on leveraging the object detection capabilities of YOLO to enhance the efficiency of traffic movement and

alleviate congestion. Implementing adaptive traffic signal control algorithms that respond in real-time to detected traffic patterns could significantly improve urban mobility.

- **Multi-Modal Traffic Monitoring:** Expanding the scope of object detection to include additional modalities such as pedestrian detection, traffic sign recognition, and even weather conditions could enhance overall traffic monitoring and safety.
- **Anomaly Detection:** Developing anomaly detection algorithms that can identify unusual traffic events, accidents, or security threats could further enhance the utility of the YOLO model in urban surveillance.
- **Scalability and Robustness:** Investigating the scalability of the model to handle larger intersections and assessing its robustness under adverse weather conditions, low light, or occlusions would be critical for practical utility.

# REFERENCES


[1] B. A. Alpatov, P. V. Babayan and M. D. Ershov, "Vehicle detection and counting system for real-time traffic surveillance," 2018 7th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 2018, pp. 1-4, doi: 10.1109/MECO.2018.8406017.

[2] C. Chen, “Edge intelligence empowered vehicle detection and image segmentation for autonomous vehicles,” IEEE Transactions on Intelligent Transportation Systems, 2023.

[3] C. Pornpanomchai, T. Liamsanguan, and V. Vannakosit, “Vehicle detection and counting from a video frame,” in 2008 International Conference on Wavelet Analysis and Pattern Recognition, 2008.

[4] C. Priemer and B. Friedrich, "A decentralized adaptive traffic signal control using V2I communication data," 2009 12th International IEEE Conference on Intelligent Transportation Systems, St. Louis, MO, USA, 2009, pp. 1-6, doi: 10.1109/ITSC.2009.5309870.

[5] Cai, Chen, Chi Kwong Wong, and Benjamin G. Heydecker. "Adaptive traffic signal control using approximate dynamic programming." Transportation Research Part C: Emerging Technologies 17.5 (2009): 456-474.

[6] Chabchoub, Abdelkader, et al. "Intelligent traffic light controller using fuzzy logic and image processing." International Journal of Advanced Computer Science and Applications 12.4 (2021): 396-399.

[7] D. Padilla Carrasco, H. A. Rashwan, M. Á. García, and D. Puig, “T-YOLO: Tiny vehicle detection based on YOLO and multi-scale convolutional neural networks,” IEEE Access, vol. 11, pp. 22430–22440, 2023.

[8] E. Avşar and Y. Ö. Avşar, “Moving vehicle detection and tracking at roundabouts using deep learning with trajectory union,” Multimed. Tools Appl., vol. 81, no. 5, pp. 6653–6680, 2022.

[9] Essa, Mohamed, and Tarek Sayed. "Self-learning adaptive traffic signal control for real-time safety optimization." Accident Analysis & Prevention 146 (2020): 105713.

[10] H. Ge, Y. Song, C. Wu, J. Ren, and G. Tan, “Cooperative deep Q-learning with Q-value transfer for multi-intersection signal control,” IEEE Access, vol. 7, pp. 40797–40809, 2019.

[11] H. Song, H. Liang, H. Li, Z. Dai, and X. Yun, “Vision-based vehicle detection and counting system using deep learning in highway scenes,” Eur. Transp. Res. Rev., vol. 11, no. 1, 2019.

[12] H. Tayara, K. Gil Soo, and K. T. Chong, “Vehicle detection and counting in high-resolution aerial images using convolutional regression neural network,” IEEE Access, vol. 6, pp. 2220–2230, 2018.

[13] H. Wang, Y. Yu, Y. Cai, X. Chen, L. Chen, and Q. Liu, “A comparative study of state-of-the-art deep learning algorithms for vehicle detection,” IEEE Intell. Transp. Syst. Mag., vol. 11, no. 2, pp. 82–95, Summer 2019.

[14] J. Gleason, A. V. Nefian, X. Bouyssounousse, T. Fong and G. Bebis, "Vehicle detection from aerial imagery," 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 2011, pp. 2065-2070, doi: 10.1109/ICRA.2011.5979853.

[15] K. Liu and G. Mattyus, "Fast Multiclass Vehicle Detection on Aerial Images," in IEEE Geoscience and Remote Sensing Letters, vol. 12, no. 9, pp. 1938-1942, Sept. 2015, doi: 10.1109/LGRS.2015.2439517.

[16] K. Pandit, D. Ghosal, H. M. Zhang and C. -N. Chuah, "Adaptive Traffic Signal Control With Vehicular Ad hoc Networks," in IEEE Transactions on Vehicular Technology, vol. 62, no. 4, pp. 1459-1471, May 2013, doi: 10.1109/TVT.2013.2241460.

[17] L. W. Sommer, T. Schuchert and J. Beyerer, "Fast Deep Vehicle Detection in Aerial Images," 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA, 2017, pp. 311-319, doi: 10.1109/WACV.2017.41.

[18] M. A. Zuraimi and F. H. K. Bin, “Vehicle detection and tracking using YOLO and DeepSORT,” in 2021 IEEE 11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), IEEE, 2021.

[19] M. B. Jensen, A. Mogelmose, and T. B. Moeslund, “Presenting the multi-view traffic intersection dataset (MTID): A detailed traffic-surveillance dataset,” in 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 2020.

[20] M. Hassaballah, M. A. Kenk, K. Muhammad, and S. Minaee, “Vehicle detection and tracking in adverse weather using a deep learning framework,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 7, pp. 4230–4242, 2021.
[21] M. Tajalli, M. Mehrabipour, and A. Hajbabaie, “Network-level coordinated speed optimization and traffic light control for connected and automated vehicles,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 11, pp. 6748–6759, 2021.
[22] M. Y. Yang, W. Liao, X. Li and B. Rosenhahn, "Deep Learning for Vehicle Detection in Aerial Images," 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 2018, pp. 3079-3083, doi: 10.1109/ICIP.2018.8451454.
[23] McKenney, Dave, and Tony White. "Distributed and adaptive traffic signal control within a realistic traffic simulation." Engineering Applications of Artificial Intelligence 26.1 (2013): 574-583.
[24] N. Miller, M. A. Thomas, J. A. Eichel, and A. Mishra, “A hidden Markov model for vehicle detection and counting,” in 2015 12th Conference on Computer and Robot Vision, 2015.
[25] N. Seenouvong, U. Watchareeruetai, C. Nuthong, K. Khongsomboon, and N. Ohnishi, “A computer vision based vehicle detection and counting system,” in 2016 8th International Conference on Knowledge and Smart Technology (KST), 2016.
[26] N. Zarei, P. Moallem, and M. Shams, “Fast-Yolo-rec: Incorporating Yolo-base detection and recurrent-base prediction networks for fast vehicle detection in consecutive images,” IEEE Access, vol. 10, pp. 120592–120605, 2022.
[27] O. O. Khalifa, “Vehicle detection for vision-based intelligent transportation systems using convolutional neural network algorithm,” Journal of Advanced Transportation, vol. 2022, pp. 1–11, 2022.
[28] P. Premaratne, I. Jawad Kadhim, R. Blacklidge, and M. Lee, “Comprehensive review on vehicle detection, classification and counting on highways,” Neurocomputing, no. 126627, p. 126627, 2023.
[29] Razakarivony, Sebastien, and Frederic Jurie. "Vehicle detection in aerial imagery: A small target detection benchmark." Journal of Visual Communication and Image Representation 34 (2016): 187-203.
[30] S. Chadwick, W. Maddern, and P. Newman, “Distant vehicle detection using radar and vision,” in 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 8311–8317.
[31] S. Chen and W. Lin, 2019 IEEE 3rd advanced information management, communications, electronic and automation control conference (IMCEC). IEEE, 2019.
[32] Smith, Stephen, et al. "Smart urban signal networks: Initial application of the surtrac adaptive traffic signal control system." Proceedings of the International Conference on Automated Planning and Scheduling. Vol. 23. 2013.
[33] T. Wu et al., “Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks,” IEEE Trans. Veh. Technol., vol. 69, no. 8, pp. 8243–8256, 2020.
[34] V. Gorodokin, S. Zhankaziev, E. Shepeleva, K. Magdin, and S. Evtyukov, “Optimization of adaptive traffic light control modes based on machine vision,” Transp. Res. Procedia, vol. 57, pp. 241–249, 2021.
[35] X. Dong, S. Yan, and C. Duan, “A lightweight vehicles detection network model based on YOLOv5,” Eng. Appl. Artif. Intell., vol. 113, no. 104914, p. 104914, 2022.
[36] X. Liang, X. Du, G. Wang, and Z. Han, “A deep reinforcement learning network for traffic light cycle control,” IEEE Trans. Veh. Technol., vol. 68, no. 2, pp. 1243–1253, 2019.
[37] X. Wang, L. Ke, Z. Qiao, and X. Chai, “Large-scale traffic signal control using a novel multiagent reinforcement learning,” IEEE Trans. Cybern., vol. 51, no. 1, pp. 174–187, 2021.
[38] Y. Miao, F. Liu, T. Hou, L. Liu, and Y. Liu, “A nighttime vehicle detection method based on YOLO v3,” in 2020 Chinese Automation Congress (CAC), 2020.
[39] Z. Rahman, A. M. Ami and M. A. Ullah, "A Real-Time Wrong-Way Vehicle Detection Based on YOLO and Centroid Tracking," 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 2020, pp. 916-920, doi: 10.1109/TENSYMP50017.2020.9230463.

[40] Z. Zheng et al., "A Novel Vehicle Detection Method With High Resolution Highway Aerial Image," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 6, no. 6, pp. 2338-2343, Dec. 2013, doi: 10.1109/JSTARS.2013.2266131.

[41] Vedansh Bhardwaj, Yaswanth Rasamsetti, Vipina Valsan, Image Processing Based Smart Traffic Control System for Smart City, 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), IEEE.

[42] Sanjai Siddharthan, M., S. Aravind, and S. Sountharrajan. "Real-Time Road Hazard Classification Using Object Detection with Deep Learning." International Conference on IoT Based Control Networks and Intelligent Systems. Singapore: Springer Nature Singapore, 2023.

[43] Raj, V. Sowbaranic, et al. "Smart Traffic Control for Emergency Vehicles Prioritization using Video and Audio Processing." 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, 2022.

[44] Tarachandy, Sahul Mohan, and J. Aravinth. "Enhanced local features using ridgelet filters for traffic sign detection and recognition." 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, 2021.