Available online at www.ijournalse.org

**Emerging Science Journal**

(ISSN: 2610-9182)

Vol. 9, No. 2, April, 2025

# Real-Time Vehicle Type Detection and Counting for Emission Pollution Monitoring and Traffic Violation Identification

Md Mizanur Rahman [1], Mohammad Asifur Rahim [1], Umme Ayman [1], Amir Sohel [1], Md. Ali Hossain [1*], Mohammad Ali Moni [2, 3], Ahmed Moustafa [4, 5*]

[1] *Faculty of Science and Information Technology, Daffodil International University, Birulia, Savar, 1216 Dhaka, Bangladesh.*

[2] *Artificial Intelligence and Cyber Futures Institute, Charles Sturt University, Bathurst 2795, Australia.*

[3] *Rural Health Research Institute, Charles Sturt University, Orange 2800, Australia.*

[4] *Department of Human Anatomy and Physiology, The Faculty of Health Sciences, University of Johannesburg, Johannesburg, South Africa.*

[5] *Centre for Data Analytics and School of Psychology, Bond University, Gold Coast, Queensland, Australia.*

**Abstract**

In an urban environment, traffic congestion, automobile emissions, and road safety are significant problems that lead to financial losses and environmental deterioration. Vehicle detection, counting, speed estimation, and pollution monitoring are frequently handled inconsistently by current traffic monitoring systems. This research closes this gap by applying the cutting-edge YOLOv8 deep learning architecture to create an extensive real-time vehicle recognition and counting system. In addition to calculating vehicle speeds and detecting five different types of vehicles and pedestrians, the system also estimates emission rates in real time using traffic data. After evaluation of two YOLOv8 variants (YOLOv8n and YOLOv8s), it was found that YOLOv8s performed better, with 0.936 precision, 0.822 recall, and 0.930 mAP50 for CNG automobiles. With an emission factor ranging from 0.6 to 0.8, real-time pollution monitoring was made possible by calculating vehicle emissions based on both type and speed. In addition, the system has a web application developed with the Flask framework and allows real-time traffic data display, including emission rates, vehicle counts, and speed calculations. The method is effective, as evidenced by the results, where YOLOv8s exceed YOLOv8n in essential metrics, including the miss-classification rate (as low as 0.112) and F1-score (0.875 for CNG). With its unique method of simultaneous vehicle identification, counting, speed estimation, and pollution monitoring, this research could lead to advancements in road safety, traffic management, and emission reduction.

**Keywords:**

Emission Pollution;

Traffic Violation;

Vehicle Counting;

OpenCV; YOLOV8;

Object Detection; Computer Vision;

Vehicle Speed.

**Article History:**

| | | | |
|---|---|---|---|
| **Received:** | 19 | August | 2024 |
| **Revised:** | 23 | March | 2025 |
| **Accepted:** | 29 | March | 2025 |
| **Published:** | 01 | April | 2025 |

## 1- Introduction

The rapid increase in vehicle numbers and outdated traffic infrastructure have exacerbated urban traffic congestion, leading to significant societal challenges. These challenges span multiple areas, including economic inefficiencies, environmental degradation, public safety concerns, and elevated rates of traffic violations. As traffic congestion intensifies, cities face a rise in accidents, delays in tracking vehicles, increased emissions, and frequent breaches of traffic regulations. Intelligent Traffic Management Systems (ITMS) have been identified as a potential solution to address these pressing issues. These systems leverage modern technology to monitor traffic, detect vehicles in real-time, and enforce regulations to improve road safety and efficiency [1].

* **CONTACT**: ali.cse@diu.edu.bd; amoustaf@bond.edu.au

**DOI:** http://dx.doi.org/10.28991/ESJ-2025-09-02-023

Among the core functions of ITMS is vehicle detection and tracking. The ability to accurately detect, classify, and track vehicles on roadways is vital for ensuring smooth traffic flow and preventing accidents, especially those caused by speeding [2]. Recent advancements in deep learning and computer vision have revolutionized traffic monitoring. Object detection algorithms such as SSD [3], Fast R-CNN [4], and various versions of YOLO [5, 6] have enabled real-time vehicle detection with significant improvements in accuracy and speed [7]. However, while models like YOLOv7 and YOLOv8 offer substantial improvements in both performance and efficiency, few studies have fully utilized the potential of YOLOv8 for tasks like real-time vehicle detection, speed estimation, and emissions monitoring [8, 9].

The increasing number of vehicles on the road also contributes to environmental issues, mainly through heightened emissions. In response, there is growing interest in systems that monitor traffic and assess vehicle emissions in real time. Past research has explored vehicle speed estimation and emissions monitoring as separate domains. For instance, Fernández Llorca et al. [10] introduced a system that estimates vehicle speeds but lacks emissions tracking. Other approaches have employed physical setups to detect excessive emissions and notify drivers [11], while machine learning and IoT solutions have been developed for vehicle pollution monitoring [12]. However, only some studies have successfully linked emissions tracking to vehicle speed estimation, an important factor in determining emission rates. Additionally, vehicle classification and counting are crucial for monitoring traffic density and ensuring compliance with traffic regulations. Lin et al. [13] developed a real-time vehicle counting and classification system, though it lacked integration with emissions monitoring. Similarly, Shah Junayed et al. [14] employed YOLO-based detection for vehicles, but their system needed to incorporate the latest advances in deep learning, such as YOLOv8, and it addressed emissions control.

A significant research gap has been identified after reviewing the existing studies: no traffic monitoring system has been devised that can simultaneously and accurately identify, count, estimate speed, and monitor emission rates in real-time, and it needs the use of an advanced version of YOLO. This study aims to fill this gap by developing an integrated system with advanced YOLOv8 deep learning architecture that detects, classifies, and counts vehicles and monitors their emissions in real-time.

The major contributions of this study are given as follows:

- This study introduces an innovative real-time vehicle detection and counting system that employs the YOLOv8 deep learning architecture. By leveraging YOLOv8's advanced capabilities, the system achieves high accuracy in detecting and tracking vehicles across diverse traffic scenarios.
- Identifying when a car crosses a predetermined virtual line offers a dynamic method of counting cars and accurate real-time traffic flow statistics.
- This system determines each vehicle's speed in real-time with a high degree of precision, enhancing road safety. This feature enables thorough traffic situation monitoring and analysis, supporting pre-emptive steps to guarantee the safety of all road users.
- To the best of our knowledge, this is the first study to show the emission rate in real-time.
- Lastly, this work presents a real-time traffic monitoring system that can be used for simultaneous emission rate monitoring, speed estimates, vehicle detection, and counting.

### *1-1- Literature Review*

The exact focus of this part is to explore the other research efforts related to the study that we have selected. Specifically, several studies, theories, methodologies, performance analyses, and limitations of the existing studies are explored, which indicates the further research scope and excels our study in a different dimension. Eventually, this part of our study is expected to provide significant information that will direct further research endeavors.

Lin et al. [13] proposed a real-time traffic monitoring system utilizing a virtual detection zone, GMM, and YOLOv4 for efficient vehicle counting, classification, and speed estimation, achieving high accuracy in various conditions. Although the proposed method offers precise vehicle counting, classification, and speed estimation in real-time scenarios, enhancing traffic management and safety measures, the system assumes vehicles are within the virtual detection zone, necessitating a sufficiently broad zone, and future work aims at algorithm acceleration and model simplification to address this limitation. Shah Junayed et al. [14] presented a real-time front vehicle detection system utilizing a YOLO-based enhanced feature extractor, achieving high performance in detecting and classifying vehicles from images and videos. However, the system's performance is limited by the absence of the latest algorithms, such as YOLOv8, which could potentially enhance detection accuracy and speed. Azimjonov & Özmen et al. [15] proposed a methodology to significantly enhance the classification accuracy of the YOLO-based vehicle detector from 57% to 95.45% by integrating it with a selected classifier. Additionally, developing a novel bounding box-based vehicle tracking algorithm further improves the accuracy of vehicle counting tasks, demonstrating the effectiveness of the proposed approach in real-time highway traffic monitoring systems. However, the study does not address the scalability of the proposed system to handle a more extensive variety of traffic scenarios and environments. Wu et al. [16] introduced YOLO v5s-Ghost, a modification of the YOLO v5s network structure, enabling real-time vehicle detection and distance estimation in the CARLA simulation environment. Replacing BottleneckCSP with Ghost Bottleneck achieves a detection speed of 47 FPS with only a slight decrease in mAP by 2.6%, paving the way for further accuracy improvements while maintaining detection speed. However, the absence of several other metrics such as accuracy, precision, recall, F1-score, and IOU score is a significant drawback of this study.

Khazukov et al. [17] focused on improving vehicle speed and direction data extraction from street surveillance cameras. However, challenges like varying viewing angles and object overlap exist. They addressed these by enhancing YOLO v3 with a mask branch and optimized anchors. Although they introduced a real-time speed determination method using perspective transformation, other issues like overlapping objects and detailed classification were not addressed. Huang et al. [18] used the YOLOv3-DL algorithm, optimized with DIOU, which enhances traffic flow monitoring accuracy by 3.86% compared to YOLOv3, achieving 98.8% accuracy and 25 ms detection speed, but still faces challenges with complex scenarios and varying vehicle sizes.

The YOLOv3 algorithm effectively detects multiple traffic violations with high accuracy, achieving 97.67% in vehicle counting and 89.24% in speed detection, but performance varies with traffic density. Franklin & Mohana [19] proposed a system where the YOLOv3 algorithm effectively detects multiple traffic violations accurately, achieving 97.67% in vehicle counting and 89.24% in speed detection, but performance varies with traffic density. However, the system's operation speed decreases with higher traffic density, requiring further optimization to handle high-volume traffic efficiently. Jana et al. [20] proposed a fully automated vehicle trajectory classification method at intersections using a new similarity measure and stop bar identification, achieving high accuracy without manual intervention. However, the proposed method is limited to single intersection approaches and susceptible to tracking inaccuracies, especially in challenging conditions like nighttime or heavy occlusion. Kim et al. [21] developed a classification method to detect parking violations using camera images, achieving a 93.9% detection ratio with an average execution time of 0.367 seconds by analyzing vehicle shadows and lane types. However, the method lacks an algorithm for detecting lanes from video datasets, limiting its application to still images.

Haque et al. [22] proposed a computer vision-based lane detection system utilizing preprocessing, thresholding, and perspective transforms, which efficiently detects lanes under various environmental conditions for implementation in Advanced Driver Assistance Systems. However, the system currently requires well-marked lanes and lacks real-time hardware implementation, limiting immediate practical use in dynamic driving conditions. Patira [23] employed computer vision where YOLOV3 was used for detecting over-speeding from car dashboards, achieving about 90% accuracy in daylight with potential expansion to traffic light violations. However, the system requires high computational power for inference, with reduced accuracy in night conditions, and may incorrectly report traffic light violations from a distance. Wu & Dong [24] proposed a network called YOLO-SE, enhanced with SEF and SPPFE modules, attention mechanisms, and transformer prediction heads, significantly improving multi-scale and small-object detection in remote sensing images, achieving a mAP of 86.5%. However, there needs to be a focus on optimizing network complexity and expanding its application while addressing deployment challenges associated with the new features. Zhao et al. [25] presented an enhanced YOLOv8 algorithm that incorporates data augmentation and an AFPN for feature fusion, significantly reducing missed detections and increasing the accuracy of traffic sign detection. However, the study did not address the computational requirements or optimization strategies necessary to ensure real-time performance in traffic sign detection, potentially hindering practical deployment in real-world scenarios.

Li & Wang [26] presented a highly accurate and fast traffic sign recognition system using a fusion of faster R-CNN and MobileNet architectures, enhanced by color and shape analysis and an asymmetric kernel CNN for adequate classification. However, despite its effectiveness, the proposed system's performance under extreme weather conditions or with heavily occluded signs was not explicitly addressed. Hu et al. [27] introduced a model called SINet, a scale-insensitive convolutional neural network that utilizes context-aware RoI pooling and a multi-branch decision network to efficiently detect vehicles of varying sizes with high accuracy and speed, achieving up to 37 FPS on diverse datasets. However, they did not thoroughly discuss how the new techniques might slow down or complicate existing advanced network systems when used in real-time situations. Dong et al. [28] introduced an unsupervised convolutional neural network pre-trained with sparse filtering and a layer-skipping strategy for effective vehicle type classification from frontal view images, using $soft_{max}$ regression for final classification. However, the paper did not address the model's inability to detect vehicles from perspectives other than the frontal view, potentially limiting its effectiveness in diverse real-world scenarios. Liu et al. [29] enhanced legacy traffic camera systems with state-of-the-art computer vision techniques to accurately detect, classify, and measure vehicles, pedestrians, and cyclists, providing high-quality traffic data comparable to modern sensors. However, they did not use the latest computer vision or deep learning algorithms. Alqaraghul & Ata [30] introduced a vehicle detection technique using optimized YOLOv3 and YOLOv4 neural networks, achieving up to 99.68% accuracy and 91% precision, outperforming previous algorithms in both speed and detection capabilities. However, YOLOv4 struggles with detecting distant or variably-sized vehicles, impacting the accuracy of traffic flow predictions and statistics.

Table 1 summarizes existing research works in terms of applied methodology and research contexts. From the above, most existing studies have applied traditional approaches with CNN and conventional YOLO models, and fewer studies have applied YOLOv8 or an updated form of YOLOv8. On the other hand, most of the existing studies have focused on vehicle detection, variable-sized vehicle detection, traffic sign detection, lane detection, and traffic flow monitoring, and very few studies have focused on other crucial factors such as vehicle counting, speed estimation. Notably, no study has focused on emission rate monitoring. Our study focuses on a real-time vehicle detection system with practical and optimal emission rate monitoring, vehicle counting, and speed estimation by utilizing the YOLOv8 deep learning architecture.

**Table 1. Summary Previous Study Analysis**

| Author info | Studies (Write about implement methodology) | Analysis (Limitation or focus point) |
|---|---|---|
| Lin et al. (2021) [13] | Gaussian mixture model (GMM), and YOLOv4 | vehicle counting, classification, and speed estimation |
| Shah Junayed et al. (2021) [14] | Traditional YOLO | Detecting and classifying vehicles |
| Azimjonov & Özmen (2021) [15] | YOLO. CNN based classifiers, bounding box bases tracker | Vehicle counting, real-time highway traffic monitoring system |
| Wu et al. (2021) [16] | YOLOv5s-Ghost | Vehicle detection and estimation |
| Khazukov et al. (2020) [17] | YOLOv3 | Vehicle speed and Vehicle direction data extraction |
| Huang et al. (2020) [18] | YOLOv3-DL | Traffic flow monitoring |
| Franklin & Mohana (2020) [19] | YOLOv3 | Detection of multiple traffic violations |
| Jana et al. (2023) [20] | Unsupervised hierarchical clustering technique | Vehicle trajectory classifications |
| Kim et al. (2016) [21] | Using GPS tracker | Lane detection for parking violations |
| Haque et al. (2019) [22] | Gradient and HLS thresholding | Lane detection |
| Patira (2019) [23] | YOLOv3 | Detection of over speed |
| Wu and Dong (2023) [24] | YOLO-SE –YOLOv8 based | Multi-scale object detection |
| Zhao et al. (2023) [25] | YOLOv8 | Traffic sign detection |
| Li and Wang (2018) [26] | R-CNN and MobileNet | Traffic sign recognition |
| Hu et al. (2018) [27] | SINet | Detection of varying sizes vehicle with high accuracy and speed |
| Dong et al. (2015) [28] | Unsupervised CNN | Vehicle type detection |
| Liu et al. (2020) [29] | CNN with projective geometry information | detect, classify, and measure vehicles, pedestrians, and cyclists |
| Alqaraghul et al. (2022) [30] | YOLOv3 and YOLOv4 | Detection of variable-sized vehicles with traffic flow predictions |

## 2- Proposed Methodology

The approach for real-time vehicle object recognition and monitoring from the standpoints of traffic violations and environmental pollution is presented in this paper. We have attempted to incorporate an extraordinary and vital thought that does not just concentrate on one issue from the very beginning of this investigation. As we previously stated, there has not been any previous research on this emission rate computation to the best of our knowledge. When we concentrate on this problem, we discover that several tasks can be completed in a single system, improving our neighborhood's traffic and environmental conditions. There is no way for us to avoid using vehicles for transportation in our everyday lives. However, we are aware that many traffic-related issues, such as gas and gasoline for vehicles, or other factors, particularly the carbon issue, result in significant pollution from transportation.

The approach depicted in Figure 1 might be the best for this study and any subsequent research to enhance our recommendation's precision and dynamic quality. Figure 1 depicts the complete study process, with distinct subsections delving deeply into each stage. Clearly, our inquiry heavily relies on the methodological flow chart for the linear answer we found.

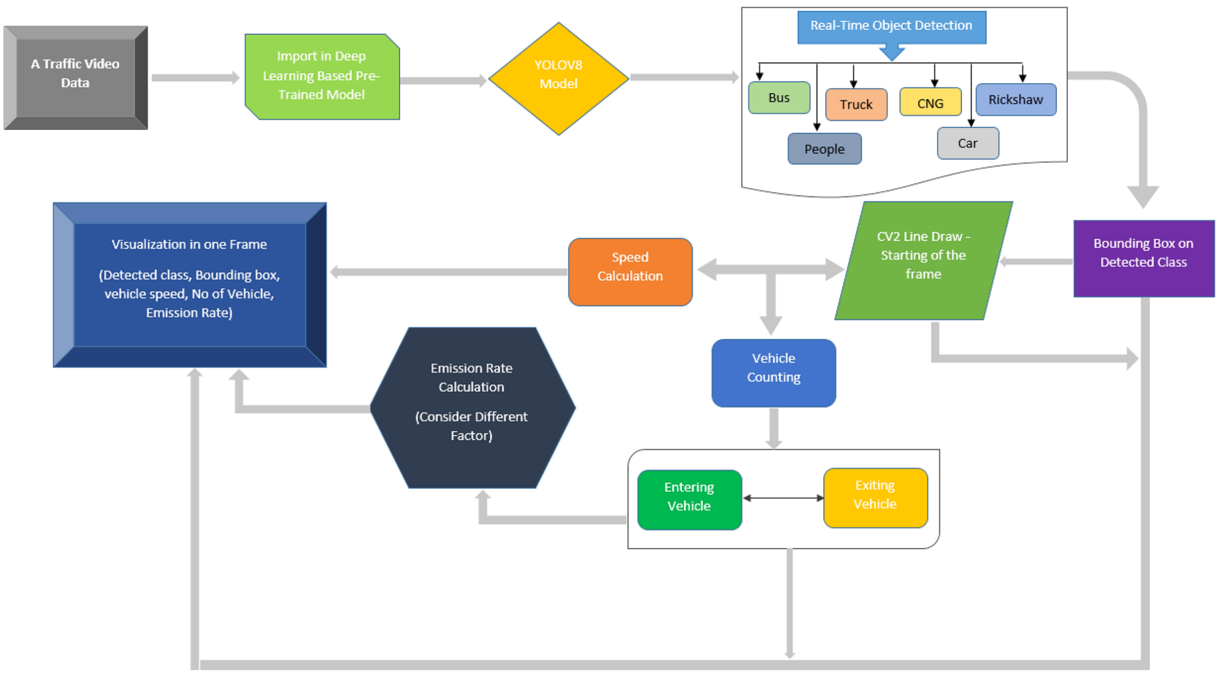



**Figure 1. Methodological Diagram**

## 3- Model Description

YOLO (You Only Look Once) is an effective technique for object identification. YOLOv8, an improved version of YOLO, uses a Convolutional Neural Network (CNN) [31] to improve object detection accuracy. The input image is segmented into a grid, and the YOLOv8 method, in contrast to dual-stage models like Faster-RCNN [32], promises quicker detection at the sacrifice of some degree of accuracy. Bounding boxes and class probabilities are predicted for each grid cell.

YOLOv8 is a powerful system composed of three essential parts [33]: the head, neck, and backbone, as illustrated in Figure 2. The backbone network, serving as the central component of YOLOv8, is meticulously designed with the combination of CBS, C2F, and SPPF modules. Each component of YOLOv8 works in harmony, transforming the underlying architecture against the traditional top-down up sampling of PAN-FAN [34]. As YOLOv8's core part, the backbone gives the network unprecedented vitality and agility by fully exploiting the gradient-enriched C2f impact. This novel method improves performance and establishes the foundation for object detection that is more reliable and effective, showcasing the strength of YOLOv8's collaborative design.

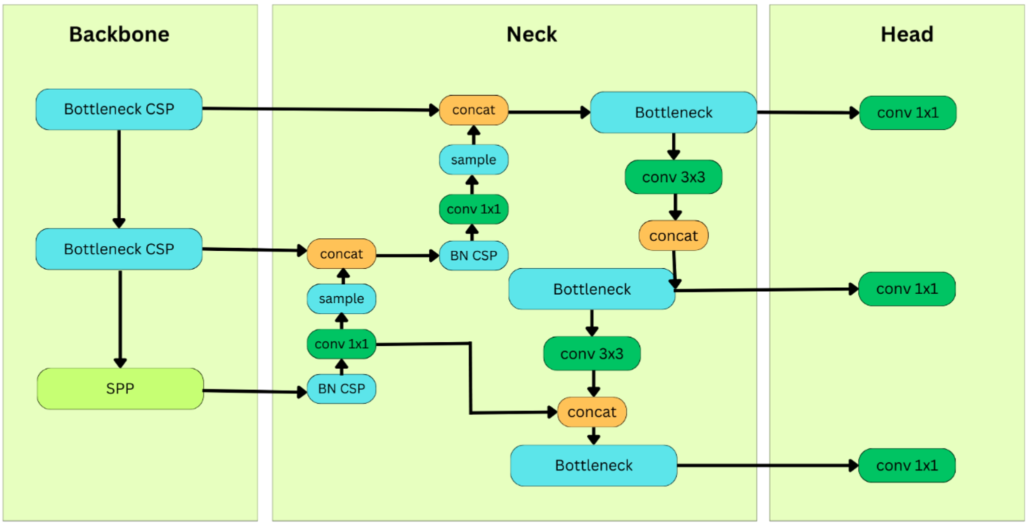



**Figure 2. YOLOv8 components**

By eliminating the limitations of the conventional C3 module, YOLOv8 [35] ushers in a new era of efficacy and efficiency. It encounters the neck layer, an essential component that alters the basic structure of the network's connections. The updated neck layer facilitates smooth gradient flow, which enhances the network's information transmission capabilities. Ultimately, this innovation breathes new life into the detection approach by enabling YOLOv8 to capture minute details with previously unheard-of accuracy.

The head network has finally emerged by breaking from the conventional methods and separating detection and classification through a transformational process, which is the apex of YOLOv8's ingenuity and innovation. YOLOv8's adoption of an anchor-free methodology [24] not only represents a significant advancement in the field of object identification algorithms, but it also maximizes computing resources and accelerates convergence by allowing each object to function independently.

Eventually, YOLOv8's output outlines the separation of detection and classification, using top-level features for large targets and bottom-level characteristics for smaller ones, creating a vector with category and position information [25].

Grid cells, bounding boxes, classes, and input video or image are used by the YOLOv8 algorithm. The original video or picture that is given into the object detection algorithm [36] is known as the input video or image. A grid of cells is created from the supplied image [37]. Every cell's job is to find items in its own spatial domain. Similar to YOLOv5 [38], YOLOv8 predicts bounding boxes for each grid cell that might contain objects. The center coordinates (x, y), width (w), and height (h) of these bounding boxes are shown in relation to the grid cell and the entire image. YOLOv8 can identify a variety of object classes [39]. The algorithm assigns a probability to each class for every predicted bounding box. For each grid cell, YOLOv8 predicts a set of parameters: (x, y, w, h, confidence, P(class_1), ..., P(class_C)). Where,

- (x, y) represent the center of the bounding box relative to the grid cell.
- (w, h) denote the width and height of the bounding box relative to the entire image.
- (confidence) indicates the confidence score of the predicted bounding box.
- (P(class_i)) represents the probability of the object belonging to class i among C different classes.

## 4- Implementation

### *4-1- Real Time Object Detection*

The environment is set up for real-time object detection by utilizing the YOLOv8 [35], and all necessary libraries and packages are installed for video processing with OpenCV and numerical operations with NumPy [40]. In addition, the pertinent configuration files required for model loading are obtained along with the pre-trained YOLOv8 model version, YOLOv8 [35]. Once the environment is prepared, the YOLOv8 pre-trained model is loaded along with the configuration files, and this process ensures that the model is prepared to process incoming video frames for the object detection task, where each frame of video streaming is processed by utilizing OpenCV(cv2) [41], as well as this involves reading each frame, preparing it for inference, and passing it through the YOLOv8 model. During the object detection phase, the YOLOv8 model predicts bounding boxes [38] and class labels for each item found inside the video frames. These predictions provide critical information about the items in the scene.

After obtaining the object detection results, the recognized items are listed in Table 2 after being filtered from the video stream based on the classes of interest, which include individuals, trucks, buses, rickshaws, and CNG vehicles. Bounding boxes are drawn around the detected items on each frame in order to visualize the object detection findings, and this procedure aids in the visual verification of the object identification process' correctness. In order to accomplish real-time [42] object recognition and to ensure that the YOLOv8 model detects and tracks things in real-time, the processing process is continuously repeated for each frame in the video stream. This continuous processing, without any breaks, provides us with instant insights into the video stream.

**Table 2. Name of target classes**

| Sl. No. | Class Name |
|---|---|
| 1 | Bus |
| 2 | Truck |
| 3 | Rickshaw |
| 4 | CNG |
| 5 | People |
| 6 | Car (default class) |

At the time of model implementation, we must create a bounding box into the detected class (Figure 3). To evaluate the model, we can measure Intersection over Union (IoU) if there are any failures. The Intersection over Union (IoU) is a crucial metric in object detection, including YOLOv8. It is calculated as the ratio of the area of overlap between a predicted bounding box and a ground truth bounding box to the area of their union. The IoU ranges from 0 to 1, where 1 indicates a perfect overlap and 0 indicates no overlap at all. In YOLOv8, IoU is utilized to evaluate the accuracy of object detection predictions, particularly in non-maximum suppression, where overlapping bounding boxes are filtered to retain the most accurate detections.


**Figure 3. Bounding box on detected class**

### *4-2- Cv2 Line Draw & Counting Entering & Exiting Car*

After object detection is complete, the focus shifts to vehicle counting. To do this, a line [20] must be defined within the frame, as shown in Figure 4, to determine whether a vehicle is entering or exiting. Each vehicle that is detected is then tracked, and the model determines which direction it crosses the line.


**Figure 4.** **Defined cv2 line**

The center of each vehicle's bottom edge is calculated for direction evaluation utilizing the bounding boxes [43] that were obtained during object detection. The model tracks the cars as they cross the marked line by using their movement direction (East, West, North and South). An automobile moving "North" is regarded as an entry, whereas an automobile moving "South" is regarded as an exit. Once the entries and exits have been totalled, the results are displayed on the frame. Distance measurement algorithms can be used to increase counting accuracy [44].

$$d = \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)} \tag{1}$$

where $(x_1, y_1)$ and $(x_2, y_2)$ represent the x and y coordinates of two points.

This formula calculates the distance between two points, aiding in accurately determining the movement of vehicles across the defined line. Total number of vehicles in a frame is measured by the following equation:

$$Total\ vehicle = \sum_{i=0}^{class} V_c \quad in\ each\ frame \tag{2}$$

where, $V_c$ stands for vehicle count in each frame.

The proposed system produces a visualization after the counting is complete, with the total number of entrances presented in the top right corner (as shown in Figure 5) and the total number of exits displayed in the top left corner (as shown in Figure 6).

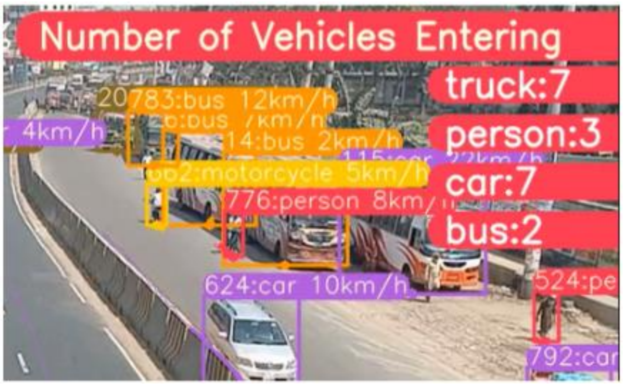



**Figure 5.** **Vehicle entry**


**Figure 6.** **Vehicle leave**

This counting procedure is essential for figuring out the emission rate [45] over a given region as well as the procedures are interconnected since the number of cars arriving and departing during a specified interval is required to calculate the emission rate.

### 4-3- Speed Calculation of Vehicle

By identifying and resolving infractions pertaining to vehicle speed [46, 47], lane adherence [22, 48], and traffic signal infringement, the proposed system's designation improves road safety. Initially, a precise method is employed to gauge a vehicle's speed as Figure 1 illustrates that entails utilizing pixel coordinates to precisely calculate the distance travelled, then using the Euclidean distance [20] algorithm to transform this data into meters.

$$d_{pixel} = \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)} \tag{3}$$

where, $d_{pixel}$ represents the distance in pixels between two points. $x_1, y_1$ represents the coordinates of the first point, $x_2, y_2$ represents the coordinates of the second point.

Through further conversion utilizing a predefined pixels-per-meter (PPM) ratio for calculate distance in meter:

$$d_{meter} = \frac{d_{pixel}}{ppm} \tag{4}$$

Here, $d_{meter}$ represents the distance in meter unit, $ppm$ represents pixels-per-meter.

And a time constant derived from the frame rate, we ascertain the vehicle's speed in kilometres per hour:

$$speed = d_{meter} \,.\, t_c \tag{5}$$

Here, the time constant $t_c$ = (15 × 3.6).

This methodical procedure guarantees a precise evaluation of a car's speed, making it easier to spot any speeding infractions. The proposed system uses a conclusive evaluation process after the speed measurement to evaluate if the observed speed is higher than the posted speed limit [23, 49] for that particular region. When the computed speed exceeds the predetermined limit, the system immediately reports the car for a speeding violation, making it possible for traffic laws to be enforced quickly and efficiently (Figure 7).


**Figure 7. Speed calculation**

Furthermore, the system under consideration enhances the impact of surveillance by focusing on lane adherence violations [21]. It can precisely identify instances in which vehicles stray from designated lanes, and in the case of a lane adherence violation, it appears as though the system would identify a bus traveling on a lane that is designated for trucks only. Furthermore, the proposed system can identify traffic signals and determine whether a vehicle is committing a traffic signal violation [50, 51]. If a violation is detected, it is managed to promptly take appropriate action to ensure compliance with traffic regulations and promote road safety.

### 4-4- Emission Pollution Rate Calculation

Early in the assigned system, video input is the foundation for additional analysis. The YOLOv8 [35] model, which aids in identifying the autos and offers bounding boxes so that their existence may be ascertained, is meticulously checked in each frame. The system counts the cars as well as from the expected bounding boxed output after the data is generated. This allows for counting car arrivals and departures during a predetermined period [52]. The most crucial information is obtained to determine emission factors ($CO_2$/km) depending on the kind of vehicle, size (weight), fuel type, age (years), and engine (CC) for a particular vehicle type, as indicated in Table 3. This table is a very analytical component of the study, and we attempt to get an average value based on multiple studies [35]. Numerous factors, particularly these four characteristics, affect the emission rate's value. Following analysis of this emission value, we solely employ the rate ranging from 0 to 1.

**Table 3. Emission factors of different classes of vehicles**

| Vehicle Type | weight | Fuel Type | Use Time of Vehicle | Engine CC | Emission Factor ($CO_2$/km) |
|---|---|---|---|---|---|
| Bus | 10000-18000KG | Diesel | 5-10 Years | 5000-12000CC | 500-1200 g |
| CNG | 500-1300kg | CNG | 5-10 years | 800-1500 | 90-150 g |
| Truck | 3500-12000kg | Diesel | 5-10 years | 4000-15000CC | 700-1300 g |
| Rickshaw | 150-400Kg | No/Petrol/CNG | 1-7years | 0- 250CC | 0-70 g |
| Others (default) | 300-1200kg | Diesel/Petrol | 1-10 years | 100-1200CC | 100-350 g |

The system enters the crucial stage of the emission rate [53, 54] computation when the counting procedure is finished. The following equation is used by the system to calculate the estimated emission rate output [55] for each frame, based on a present emission factor from Table 1:

$$ER = V_t \times E_{Factor} \% \tag{6}$$

In this case, stands for the emission factor, the total number of vehicles in each frame, and the emission rate. This table's analysis shows that the value results in a minimum of 0 and a maximum of 1. Therefore, we use the factor value of 0.6-0.8 for the average time.

The final iteration of the proposed system reveals its findings through a computation of emission rates onto the original video, and its finding is shown in Figure 8 that exhibits the emission rate continually changes over time frame to calculate real-time [53, 56] emission pollution rate.


**Figure 8. Emission rate visualizing**

Emission rate calculations in Bangladesh can yield important information [57] about the environmental health and air quality of the nation. Through precise real-time vehicle emissions quantification, this technology can enable policymakers to undertake focused interventions aimed at lowering pollution levels and enhancing public health. It may also be used to pinpoint pollution hotspots and direct resource allocation for more potent mitigating techniques. In the end, these programs may result in improved living conditions, better air, and sustainable urban growth in Bangladesh.

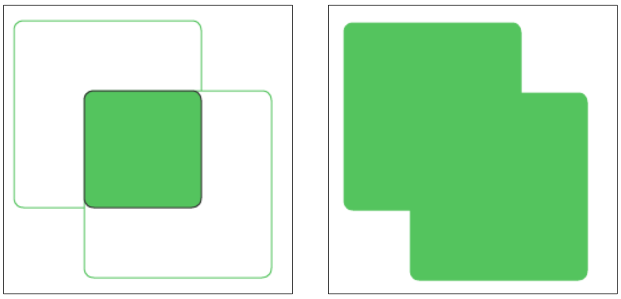
## 5- Result Analysis

All four YOLOv8 model iterations have been implemented in this study using a Tesla T4 GPU and Google Colab. The image size is 640×640, the training method consists of 50 epochs with a batch size of 8, momentum of 0.93, weight decay of 0.0005, workers of 8, pre-trained: true, optimizer of AdamW, and IoU of 0.7. The same set of parameters were used to train each model. The aforementioned configuration facilitates quick training by leveraging the GPU's processing power.

**Model Performance Metrics:** The metrics primarily applied in the field of object detection include precision, recall, intersection over union (IoU), average precision (AP), mean average precision (mAP) values, and accuracy. The IoU evaluation is applied to quantify the extent of the intersection between the bounding box predicted by the model and the ground truth bounding box in the original image [58].

$$IoU = \frac{Area\ of\ Intersection}{Area\ of\ Union} \tag{7}$$

IoU is one of the popular measurement matrices for advance deep learning techniques and here two section that is area of Intersection and Union are considered. Figure 9 can be a clear idea of this.



(a) (b)

**Figure 9. IoU (Intersection over Union): a) Area of Intersection; b) Area of Union**

The detection Result was identified as a True Positive (TP) when the IOU value calculated between the Detection Result and the Ground Truth exceeded the noted threshold value (0.7) [59].

Precision is a metric that evaluates the ratio of correctly predicted positive objects (True Positives) to the total number of objects that are predicted as positive (True Positives + False Positives) [59].

$$Precison\ = \frac{TP}{TP+FP} \tag{8}$$

The recall metric is the ratio of accurately detected objects to all positive objects in the test set [60].

$$Recall\ = \frac{TP}{TP+FN} \tag{9}$$

The precision-recall $P(r)$ curve is the graphical representation that displays precision on the vertical axis and recall on the horizontal axis. This visualization demonstrates the relationship between the classifier's accuracy in correctly finding positive instances and its capability to capture all positive instances [61].

The average precision $(AP)$ is a scalar measure that predicts the area enclosed by the precision-recall $P(r)$ curve. A higher AP value signifies the classifier's enhanced performance [62].

$$AP\ = \int_0^1 P(r)dr \tag{10}$$

where $P(r)$ is the precision at recall $(r)$.

The mAP is a numeric measure utilized to evaluate the accuracy of object detection models across all classes in a specific dataset. The formula for mAP tells us that, for a given class, $k$, we need to determine its corresponding AP [40].

$$mAP\ = \frac{1}{n}\sum_{k=1}^{k} AP_k \tag{11}$$

The harmonic mean of precision and recall is identified as $F_1 - score$ [62].

$$F_1 - score\ = \frac{2\times(Precision * Recall)}{Precision + Recall} \tag{12}$$

### *5-1- Result Analysis of Applied Models*

Our main objectives in creating this essay are to reduce traffic violations and offer a remedy for the environmental issue. Specifically, we introduce a cutting-edge, contemporary method for quantifying and monitoring emissions in this process. But first, we use a very small dataset to train a pre-trained object detection algorithm. We additionally evaluate the precision, recall, and mAP score of the YOLO V8 method. We use two different versions of YOLOV8 for object segmentation in this work. Below are the outcomes of the YOLOv8s and YOLOV8n algorithms, showcasing the novelty of our approach.

This study evaluated the performance of two YOLOv8 variants—YOLOv8n (Nano) and YOLOv8s (Small)—for real-time vehicle detection, classification, and counting. The results in Tables 4 and 5 include vital metrics such as Precision, Recall, mAP50, $F_1 - score$, and Miss Classification Error Rate across six object categories: Car, Bus, Truck, CNG, Person, and Rickshaw.

- **YOLOv8n:** YOLOv8n is the Nano variant and reached the 0.799 mAP50. Table 4 shows overall result of this version.
- **YOLOv8s:** YOLOv8s is another specialized version of v8. The overall result summary in Table 5.

Tables 4 and 5 present the performance analysis of YOLOv8n and YOLOv8s respectively regarding precision, recall, mAP50, mAP50-90, f1-score and miss classification error rate metrics. Table 4 presents the results of YOLOv8n where range of precision is 0. 835-0.93, range of recall is 0.528-0.822, range of mAP50 is 0.679-0.90, range of mAP50-90 is 0.408-0.726, range of f1-score is 0.657-0.875, and range of miss classification error rate is 0.112-0.321. With the maximum precision of 0.93 for CNG vehicles and the lowest precision of 0.835 for rickshaws, YOLOv8n attained an overall mAP50 of 0.799. Recall values for CNG and Person ranged from 0.528 to 0.822, indicating that the model performed better in identifying CNG vehicles than pedestrians. The model has trouble classifying smaller or less distinct items, such pedestrians, as evidenced by the Miss Classification Error Rate, which varied from 0.112 for CNG to 0.321 for persons. Table 5 presents the results of YOLOv8s where range of precision is 0. 825-0.936, range of recall is 0.528-0.822, range of mAP50 is 0.688-0.878, range of mAP50-90 is 0.432-0.746, range of f1-score is 0.679-0.875, and range of miss classification error rate is 0.12-0.311. In most metrics, YOLOv8s performed better than YOLOv8n, with an overall mAP50 of 0.878. Notably, it had the lowest precision (0.825) for rickshaws and the best precision (0.936) for CNG. Like YOLOv8n, recall for CNG was likewise high at 0.822, although it was more consistent across all categories. With the lowest being 0.12 for CNG and 0.311 for Person, the Miss Classification Error Rate was lowered in YOLOv8s, demonstrating its enhanced capacity to identify objects more accurately across various classes.

**Table 4.** **YOLOv8n result summary**

| Class | Precision | Recall | mAP50 | mAP50-90 | F1-score | Miss Classification Error Rate (IoU:50) |
|---|---|---|---|---|---|---|
| Default (car) | 0.902 | 0.693 | 0.799 | 0.603 | 0.784 | 0.201 |
| Bus | 0.912 | 0.717 | 0.823 | 0.639 | 0.803 | 0.177 |
| Truck | 0.921 | 0.791 | 0.878 | 0.714 | 0.854 | 0.122 |
| CNG | 0.93 | 0.822 | 0.9 | 0.726 | 0.875 | 0.112 |
| Person | 0.863 | 0.528 | 0.679 | 0.408 | 0.657 | 0.321 |
| Rickshaw | 0.835 | 0.603 | 0.723 | 0.516 | 0.705 | 0.277 |

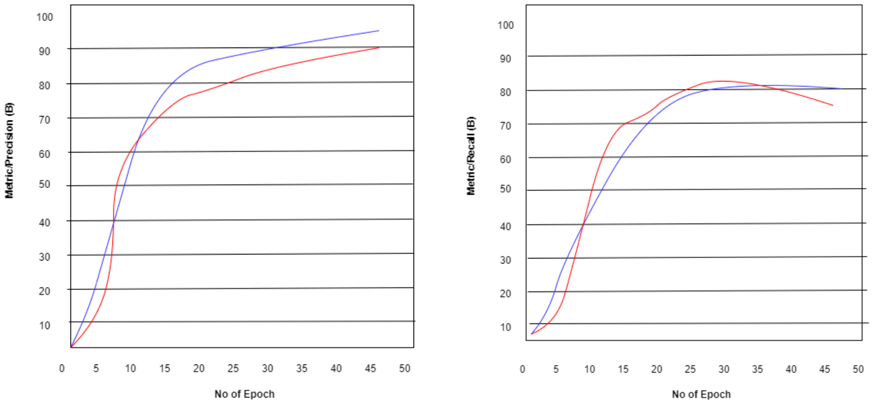
**Table 5.** **YOLOv8s result Summary**

| Class | Precision | Recall | mAP50 | mAP50-90 | F1-score | Miss Classification Error Rate (IoU:50) |
|---|---|---|---|---|---|---|
| Default (car) | 0.922 | 0.723 | 0.799 | 0.613 | 0.78 | 0.197 |
| Bus | 0.932 | 0.737 | 0.823 | 0.642 | 0.803 | 0.176 |
| Truck | 0.931 | 0.791 | 0.878 | 0.714 | 0.86 | 0.122 |
| CNG | 0.936 | 0.822 | 0.93 | 0.746 | 0.875 | 0.12 |
| Person | 0.873 | 0.528 | 0.688 | 0.432 | 0.679 | 0.311 |
| Rickshaw | 0.825 | 0.603 | 0.721 | 0.563 | 0.714 | 0.276 |

After a thorough comparison of the performance of YOLOv8n and YOLOv8s, it becomes evident that YOLOv8s consistently outperforms YOLOv8n in all metrics. This superiority instills confidence in the model's performance, especially in real-time traffic scenarios where better generalization across different Intersection-over-Union thresholds is crucial. The YOLOv8n and YOLOv8s models show higher precision and recall for CNG vehicle detection due to unique characteristics like size and shape. However, they perform worse in identifying people and rickshaws, with higher miss-classification rates and poorer recall values. YOLOv8s consistently outperforms YOLOv8n in all metrics, with an improvement in mAP50-90, indicating better generalization across different Intersection-over-Union thresholds, crucial for real-time traffic scenarios.

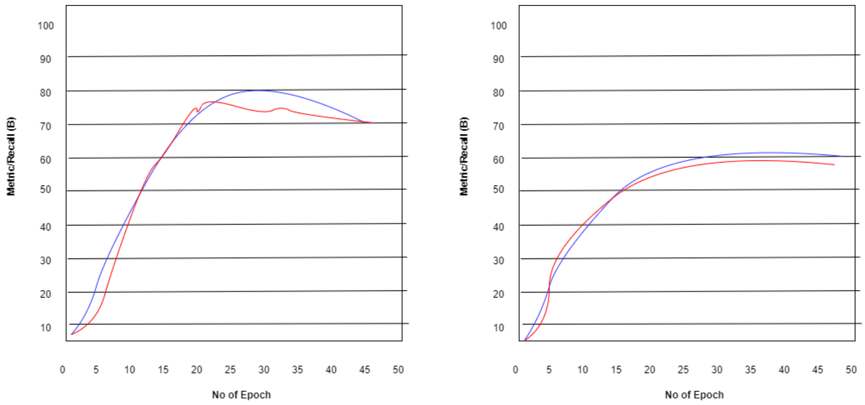
Our study, in comparison with existing studies, has employed a more advanced version of YOLO, such as YOLOv8n and YOLOv8s, rather than using the old version of the YOLO model. This choice has significantly improved the accuracy of our model. In terms of real-time vehicle detection, YOLOv8s achieves a higher performance with an overall mAP550 of 0.87 and a higher precision of 0.93 for vehicle CNG than the existing studies. This emphasis on improved accuracy reassures the audience about the capabilities of our model.

The results of our research are demonstrated in Figures 10 and 11, which is a curve visualization that focuses on four different train and validation loss graphs. We displayed the measures value-considered precision, recall into Figure 10 and mAp50(B), and mAP50-95 (B) visual in Figure 11. Red and Blue line represents the Train and Validation respectively.




(a) (b)

**Figure 10.** **Train and validation loss curve, precision, Recall, mAP50, and mAP50:90 curve**




(a) (b)

**Figure 11.** **Train and validation loss curve, precision, Recall, mAP50, and mAP50:90 curve**

## 6- Web Application Implementation

After full fill the implementation procedure we develop a real time application for better understanding and make the study more useful and creative for direct user. We firstly said that our study only proposes a unique and a very important thing for society.

### *6-1- Device Setup and Required Library for Implement Web Application*

YOLOv8 and Python Flask are used to construct a web application. Both individuals and cars may be detected, tracked, and counted by this approach. Through a web interface, the application seeks to offer a real-time car tracking, detection, and counting solution. It provides a user-friendly platform for tracking traffic flow and analysing vehicle movement trends by utilizing the Flask framework. The YOLO (You Only Look Once) object identification method is used by the system to quickly and precisely identify cars in video streams.

- **Device Configurations**: Intel core i7-4770 CPU @ 3.40GHz x64-based processor, 8 GB RAM, and Windows 10 64-bit operating system.
- **Flask Framework:** Flask, a micro web framework for Python, is the application's backbone. Its simplicity and flexibility make it an ideal choice for building web applications with Python. Flask facilitates the creation of web pages, routing, and handling of HTTP requests, enabling seamless vehicle detection and integration of tracking functionality into a web interface. By leveraging Flask, the application achieves modularity, scalability, and ease of maintenance.

- **Required Libraries:** The application utilises a range of libraries and tools to enable various functionalities:
  - PyCharm IDE: The PyCharm IDE is used for developing the web application.
  - Ultralytics: Provides the YOLO object detection model for vehicle detection.
  - NumPy: Essential for numerical computations and array manipulation.
  - Matplotlib: Utilized for data visualisation and plotting.
  - OpenCV-Python: Offers computer vision capabilities for image and video processing.
  - Flask-WTF: Simplifies form handling and validation within Flask applications.
  - CVZone: Provides additional computer vision utilities for tasks such as drawing and annotation.
  - Math: Standard mathematical functions and operations.
  - WTForms: Enables the creation and validation of web forms.
  - Werkzeug.utils: Offers utility functions for various web-related tasks.
  - WTForms.validators: Provides validators for form field validation.

### *6-2- Working Procedure Describe*

We make sure that our research decisions have a significant impact on the computer vision industry for traffic analysis through the use of cutting-edge algorithms. It is also a pleasure that our study focuses on the problem of reducing carbon emissions in order to increase the existence of our green environment and protect humans from the greenhouse effect, as emission pollution is a serious concern that affects people everywhere in the world. The web application's functioning flow diagram is shown in Figure 12.

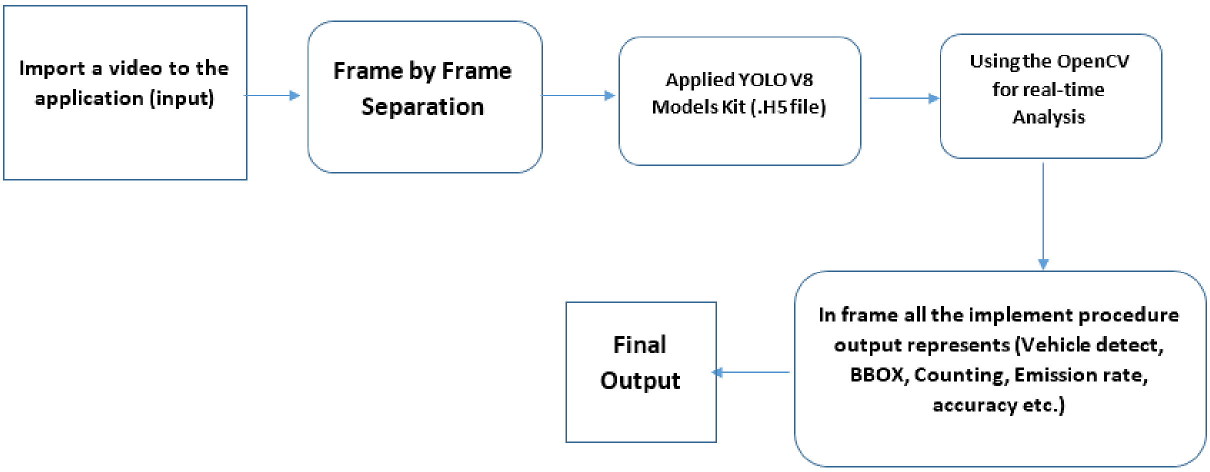



**Figure 12. Working Flow Diagram**

For Clear understanding the procedure discusses in bullet points for web application:

- Video processing: Frame by frame video input is processed.
- Object Detection: To detect vehicles, each frame is fed into the YOLO model.
- Line intersects: To identify vehicle crossings, a line is drawn in the center of each frame. A vehicle is given a distinct ID and counted when its midpoint crosses this line.
- Counting and Tracking: Cars that are detected are counted higher by being tracked between frames.
- Speed Calculation: Pre-trained Model and implemented formula help out to calculate speed of the vehicle.
- Emission: Formula ensure the emission rate with the help of vehicle tracking and counting.
- Information Display: Relevant data is annotated on each frame, including bounding boxes, class names, accuracy, FPS, counts, and tracking IDs when they cross the middle line.
- Output Generation: A final video output is created by compiling the annotated frames. The video that was uploaded is kept in a directory.

Figure 13 is a visualization of the output. It is a sample for our very primary working, and it make sure that the visual out is very helpful of the relevant responsible persons in this field.

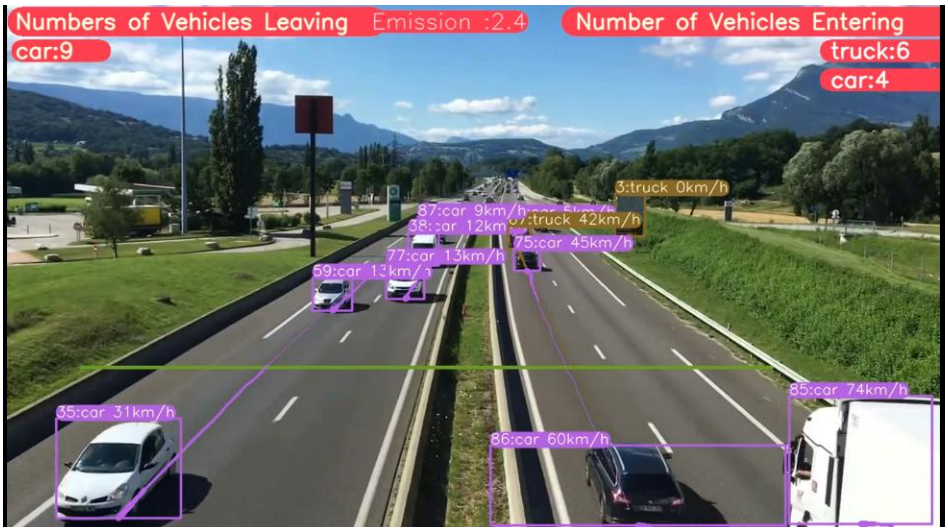



**Figure 13.** **Output Sample of web Application**

## 7- Conclusion

In conclusion, this study introduces a pioneering real-time vehicle detection and counting system using the YOLOv8 deep learning architecture. By leveraging the advanced capabilities of YOLOv8, the proposed methodology achieves high-accuracy detection and tracking of vehicles across various traffic scenarios. The system is uniquely configured to dynamically count vehicles by detecting their crossings over a predefined virtual line, providing precise real-time traffic flow analysis. To enhance road safety, the system calculates the speed of each vehicle with high precision, supporting meticulous monitoring and proactive safety measures. Notably, this study is the first to demonstrate real-time emission rate monitoring based on vehicle type, weight, engine capacity, and speed, making it one of the first to link real-time traffic data with pollution tracking.

Two variants of YOLO, YOLOv8n and YOLOv8s, were utilized in the study. The system classified six objects, including five types of vehicles and a person. The performance of both YOLOv8s and YOLOv8n is evaluated based on Precision, Recall, mAP50, mAP50-90, F1-score, and Miss Classification Error Rate(IoU:50). YOLOv8s outperformed YOLOv8n in most performance metrics, achieving a precision of 0.922 in detecting cars compared to YOLOv8n's 0.902. However, both variants exhibited lower mAP50-90 scores in detecting persons, with YOLOv8n scoring 0.408 and YOLOv8s scoring 0.432. These results highlight the effectiveness of YOLOv8s in vehicle detection while indicating areas for improvement in pedestrian detection.

Additionally, a web application has been developed to facilitate the implementation of the system. The development of this web application enhances user interaction, enabling real-time display of vehicle counts, speed, and emissions through an intuitive interface.

### *7-1- Limitations and Future Scope*

The study, despite its innovative approach, has several limitations. The real-time vehicle detection and counting system employing the YOLOv8 architecture may face challenges in varying weather and lighting conditions, which can affect detection and tracking accuracy. Additionally, the system's reliance on high-quality video input may only be feasible in some scenarios, particularly in areas with limited infrastructure. The accuracy of speed calculation and emission rate monitoring could also be impacted by factors such as occlusions, camera angles, and calibration issues. Moreover, the computational requirements for real-time processing necessitate significant hardware resources, potentially limiting the system's scalability and widespread adoption.

The future scope of this research includes several promising directions. Enhancements in robustness can be achieved by improving the system's performance under various environmental conditions, such as adverse weather and poor lighting. Integration with advanced sensor technologies and the Internet of Things (IoT) can facilitate better data acquisition and processing capabilities. Additionally, extending the system to include multi-lane and multi-intersection analysis will not only enhance its utility in more complex traffic scenarios but also open up the potential for a more comprehensive traffic analysis. The system can also be expanded to detect a wider variety of vehicles, such as bicycles, motorcycles, and trucks, for a more comprehensive traffic analysis. Furthermore, the latest advancements in deep learning, such as YOLOv9 and YOLOv10, can be incorporated to improve detection accuracy and processing speed.

## 8- Declarations

### *8-1- Author Contributions*

Conceptualization, M.M.R. and A.R.; methodology, M.M.R.; software, M.M.R. and A.R.; validation, M.M.R., A.R., A.S., and M.A.H.; formal analysis, M.M.R., M.A.H., M.A.M., and A.M.; investigation, A.S.; resources, M.M.R. and A.R; data curation, M.M.R., U.A., and A.R.; writing—original draft preparation, M.M.R., A.R., and U.A.; writing—review and editing, M.M.R., M.A.H., M.A.M., and A.M.; visualization, M.M.R.; supervision, M.A.M. and A.M.; project administration, M.M.R. and A.S; funding acquisition, A.M. All authors have read and agreed to the published version of the manuscript.

### *8-2- Data Availability Statement*

The data presented in this study are available on request from the corresponding author.

### *8-3- Funding*

The authors received no financial support for the research, authorship, and/or publication of this article.

### *8-4- Acknowledgements*

We would like to extend our heartfelt gratitude to Daffodil International University for their invaluable support and encouragement throughout this project. We also wish to thank the Bangladesh Road & Transport Corporation (BRTC) for their assistance and resources, which were crucial to the success of our research. Your guidance and collaboration have been instrumental in achieving our goals.

### *8-5- Institutional Review Board Statement*

This study was approved by the Faculty of Science and Information Technology at Daffodil International University, Bangladesh.

### *8-6- Informed Consent Statement*

An explanatory statement was given to the participants to read in which their rights and risks were outlined. After being satisfied with the statement, they signed the informed consent form.

### *8-7- Conflicts of Interest*

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

## 9- References

[1] Fazli, S. (2012). Neural Network based Vehicle Classification for Intelligent Traffic Control. International Journal of Software Engineering & Applications, 3(3), 17–22. doi:10.5121/ijsea.2012.3302.

[2] Pattanashetty, V. B., Mane, V., Iyer, N. C., & Kore, S. (2022). Traffic Rules Violation Detection System. Information and Communication Technology for Competitive Strategies (ICTCS 2020), Lecture Notes in Networks and Systems, Springer, Singapore. doi:10.1007/978-981-16-0739-4_8.

[3] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. Computer Vision – ECCV 2016, 21–37. doi:10.1007/978-3-319-46448-0_2.

[4] Lea, C., Vidal, R., Reiter, A., Hager, G. D., Hua, G., & Jégou, H. (2016). Computer Vision—ECCV 2016 Workshops. In Proceedings of the ECCV -2016, Amsterdam, Netherlands.

[5] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), 1137–1149. doi:10.1109/TPAMI.2016.2577031.

[6] Zhang, X., & Zhu, X. (2019). Vehicle Detection in the Aerial Infrared Images via an Improved Yolov3 Network. 2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP), 372–376. doi:10.1109/siprocess.2019.8868430.

[7] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. arXiv preprint, arXiv:2004.10934. doi:10.48550/arXiv.2004.10934.

[8] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint, arXiv:1804.02767. doi:10.48550/arXiv.1804.02767.

[9] Neamah, S. B., & Karim, A. A. (2023). Real-time Traffic Monitoring System Based on Deep Learning and YOLOv8. Aro-the Scientific Journal of Koya University, 11(2), 137–150. doi:10.14500/aro.11327.

[10] Fernández Llorca, D., Hernández Martínez, A., & García Daza, I. (2021). Vision-based vehicle speed estimation: A survey. IET Intelligent Transport Systems, 15(8), 987–1005. doi:10.1049/itr2.12079.

[11] Vidhi, R., & Shrivastava, P. (2018). A review of electric vehicle lifecycle emissions and policy recommendations to increase EV penetration in India. Energies, 11(3), 483. doi:10.3390/en11030483.

[12] Wang, L., Zhang, F., Pilot, E., Yu, J., Nie, C., Holdaway, J., Yang, L., Li, Y., Wang, W., Vardoulakis, S., & Krafft, T. (2018). Taking action on air pollution control in the Beijing-Tianjin-Hebei (BTH) region: Progress, challenges and opportunities. International Journal of Environmental Research and Public Health, 15(2), 306. doi:10.3390/ijerph15020306.

[13] Lin, C. J., Jeng, S. Y., & Lioa, H. W. (2021). A Real-Time Vehicle Counting, Speed Estimation, and Classification System Based on Virtual Detection Zone and YOLO. Mathematical Problems in Engineering, 2021, 1–10. doi:10.1155/2021/1577614.

[14] Shah Junayed, M., Baharul Islam, M., Sadeghzadeh, A., & Aydin, T. (2021). Real-Time YOLO-based Heterogeneous Front Vehicles Detection. 2021 International Conference on Innovations in Intelligent Systems and Applications (INISTA), 1–7. doi:10.1109/inista52262.2021.9548650.

[15] Azimjonov, J., & Özmen, A. (2021). A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways. Advanced Engineering Informatics, 50, 101393. doi:10.1016/j.aei.2021.101393.

[16] Wu, T.-H., Wang, T.-W., & Liu, Y.-Q. (2021). Real-Time Vehicle and Distance Detection Based on Improved Yolo v5 Network. 3rd World Symposium on Artificial Intelligence (WSAI-2021), 24-28. doi:10.1109/wsai51899.2021.9486316.

[17] Khazukov, K., Shepelev, V., Karpeta, T., Shabiev, S., Slobodin, I., Charbadze, I., & Alferova, I. (2020). Real-time monitoring of traffic parameters. Journal of Big Data, 7(1), 1–20. doi:10.1186/s40537-020-00358-x.

[18] Huang, Y. Q., Zheng, J. C., Sun, S. D., Yang, C. F., & Liu, J. (2020). Optimized YOLOv3 algorithm and its application in traffic flow detections. Applied Sciences (Switzerland), 10(9), 3079. doi:10.3390/app10093079.

[19] Franklin, R. J., & Mohana. (2020). Traffic Signal Violation Detection using Artificial Intelligence and Deep Learning. 2020 5th International Conference on Communication and Electronics Systems (ICCES), 839–844. doi:10.1109/icces48766.2020.9137873.

[20] Jana, U., Karmakar, J. P. Das, Chakraborty, P., Huang, T., & Sharma, A. (2023). Automated Approach for Computer Vision-Based Vehicle Movement Classification at Traffic Intersections. Future Transportation, 3(2), 708–725. doi:10.3390/futuretransp3020041.

[21] Kim, A.-R., Rhee, S.-Y., & Jang, H.-W. (2016). Lane Detection for Parking Violation Assessments. The International Journal of Fuzzy Logic and Intelligent Systems, 16(1), 13–20. doi:10.5391/ijfis.2016.16.1.13.

[22] Haque, M. R., Islam, M. M., Alam, K. S., Iqbal, H., & Shaik, M. E. (2019). A Computer Vision based Lane Detection Approach. International Journal of Image, Graphics and Signal Processing, 11(3), 27–34. doi:10.5815/ijigsp.2019.03.04.

[23] Patira, S. (2019). Over speed detection using Artificial Intelligence. M.Sc. Thesis, San Jose State University, California, United States. doi:10.31979/etd.u8qc-9d6e.

[24] Wu, T., & Dong, Y. (2023). YOLO-SE: Improved YOLOv8 for Remote Sensing Object Detection and Recognition. Applied Sciences (Switzerland), 13(24), 12977. doi:10.3390/app132412977.

[25] Zhao, D., Zhao, L., Ma, R., & Zhao, X. (2021). Research on traffic sign detection algorithm based on improved YOLOv3. Journal of Physics: Conference Series, 2026(1), 226–232. doi:10.1088/1742-6596/2026/1/012050.

[26] Li, J., & Wang, Z. (2019). Real-Time Traffic Sign Recognition Based on Efficient CNNs in the Wild. IEEE Transactions on Intelligent Transportation Systems, 20(3), 975–984. doi:10.1109/TITS.2018.2843815.

[27] Hu, X., Xu, X., Xiao, Y., Chen, H., He, S., Qin, J., & Heng, P. A. (2019). SINet: A Scale-Insensitive Convolutional Neural Network for Fast Vehicle Detection. IEEE Transactions on Intelligent Transportation Systems, 20(3), 1010–1019. doi:10.1109/TITS.2018.2838132.

[28] Dong, Z., Wu, Y., Pei, M., & Jia, Y. (2015). Vehicle Type Classification Using a Semisupervised Convolutional Neural Network. IEEE Transactions on Intelligent Transportation Systems, 16(4), 2247–2256. doi:10.1109/tits.2015.2402438.

[29] Liu, C., Huynh, D. Q., Sun, Y., Reynolds, M., & Atkinson, S. (2021). A Vision-Based Pipeline for Vehicle Counting, Speed Estimation, and Classification. IEEE Transactions on Intelligent Transportation Systems, 22(12), 7547–7560. doi:10.1109/TITS.2020.3004066.

[30] Alqaraghuli, A., & Ata, O. (2022). Optimized YOLOv4 Algorithm for Car Detection in Traffic Flow. Turkish Journal of Science and Technology, 17(2), 395–403. doi:10.55525/tjst.1123195.

[31] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data, 8(1), 1-74. doi:10.1186/s40537-021-00444-8.

[32] Wang, M., & Ren, L. (2023). SSB-YOLO: A vehicle object detection algorithm based on improved YOLOv8. Research Square (Preprint), 1-12.doi:10.21203/rs.3.rs-3743453/v1.

[33] Al Mudawi, N., Qureshi, A. M., Abdelhaq, M., Alshahrani, A., Alazeb, A., Alonazi, M., & Algarni, A. (2023). Vehicle Detection and Classification via YOLOv8 and Deep Belief Network over Aerial Image Sequences. Sustainability, 15(19), 14597. doi:10.3390/su151914597.

[34] Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path Aggregation Network for Instance Segmentation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8759-8768. doi:10.1109/cvpr.2018.00913.

[35] Kawsar, S., Biswas, S., Noor, M., & Mamun, M. S. (2023). Investigating the applicability of COPERT 5.5 emission software in Bangladesh and developing countrywide vehicular emission inventories. Environmental Science: Atmospheres, 4(1), 57–72. doi:10.1039/d3ea00047h.

[36] Gothane, D. S. (2021). A Practice for Object Detection Using YOLO Algorithm. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 7(2), 268–272. doi:10.32628/cseit217249.

[37] Pham, S. V. H., & Nguyen, K. V. T. (2023). Productivity Assessment of the Yolo V5 Model in Detecting Road Surface Damages. Applied Sciences (Switzerland), 13(22), 12445. doi:10.3390/app132212445.

[38] Rahman, R., Bin Azad, Z., & Bakhtiar Hasan, M. (2022). Densely-Populated Traffic Detection Using YOLOv5 and Non-maximum Suppression Ensembling. Proceedings of the International Conference on Big Data, IoT, and Machine Learning. Lecture Notes on Data Engineering and Communications Technologies, Springer, Singapore. doi:10.1007/978-981-16-6636-0_43.

[39] Sirisha, U., Praveen, S. P., Srinivasu, P. N., Barsocchi, P., & Bhoi, A. K. (2023). Statistical Analysis of Design Aspects of Various YOLO-Based Deep Learning Models for Object Detection. International Journal of Computational Intelligence Systems, 16(1), 126. doi:10.1007/s44196-023-00302-w.

[40] Gupta, P., & Bagchi, A. (2024). Introduction to NumPy. Essentials of Python for Artificial Intelligence and Machine Learning. Synthesis Lectures on Engineering, Science, and Technology. Springer, Cham, Switzerland. doi:10.1007/978-3-031-43725-0_4.

[41] Naveenkumar, M., & Vadivel, A. (2015). OpenCV for computer vision applications. Proceedings of national conference on big data and cloud computing (NCBDC’15), 20 March 2015, Trichy, India.

[42] Rahman, S., Rony, J. H., Uddin, J., & Samad, M. A. (2023). Real-Time Obstacle Detection with YOLOv8 in a WSN Using UAV Aerial Photography. Journal of Imaging, 9(10), 216. doi:10.3390/jimaging9100216.

[43] Yadav, A., Chaturvedi, P. K., & Rani, S. (2024). Object Detection and Tracking using YOLOv8 and DeepSORT. Advancements in Communication and Systems, 81–90, SCRS Publication, Delhi, India. doi:10.56155/978-81-955020-7-3-7.

[44] Song, H., Liang, H., Li, H., Dai, Z., & Yun, X. (2019). Vision-based vehicle detection and counting system using deep learning in highway scenes. European Transport Research Review, 11(1), 1-16. doi:10.1186/s12544-019-0390-4.

[45] Tsanakas, N. (2019). Emission estimation based on traffic models and measurements. Linköping University, Norrköping, Sweden.

[46] Biswas, K., Paul, N. K., Saha, D., Ahmed, T., & Mahmud, R. (2023). Detection of Traffic Rule Violations Using Machine Learning: An Analytical Review. Malaysian Journal of Science and Advanced Technology, 37–47. doi:10.56532/mjsat.v3i1.146.

[47] Reddy, P. S., Nishwa, T., Reddy, R. S. K., Sadviq, C., & Rithvik, K. (2021). Traffic Rules Violation Detection using Machine Learning Techniques. 2021 6th International Conference on Communication and Electronics Systems (ICCES), 1264-1268. doi:10.1109/icces51350.2021.9488998.

[48] Wydyanto, W., Nayan, N. M., Sulaiman, R., Dewi, D. A., & Kurniawan, T. B. (2024). A Hybrid Approach to Detect and Identify Text in Picture. Emerging Science Journal, 8(1), 218-238. doi:10.28991/ESJ-2024-08-01-016.

[49] Shwaly, S. A., El-Ayaat, A., & Osman, R. (2024). Assessing Multifaceted Effects of Speed Humps and Bumps: Travel Time, Safety, and Environmental Considerations. Civil Engineering Journal, 10(7), 2177-2194. doi:10.28991/CEJ-2024-010-07-07.

[50] Razak, S. F. A., Ismail, S. N. S., Bin, B. H. B., Yogarayan, S., Abdullah, M. F. A., & Kamis, N. H. (2024). Monitoring Physiological State of Drivers Using In-Vehicle Sensing of Non-Invasive Signal. Civil Engineering Journal, 10(4), 1221-1231. doi:10.28991/CEJ-2024-010-04-014.

[51] Ravish, R., Rangaswamy, S., & Char, K. (2021). Intelligent traffic violation detection. 2nd Global Conference for Advancement in Technology (GCAT-2021), 1-7. doi:10.1109/GCAT52182.2021.9587520.

[52] García-González, J., Molina-Cabello, M. A., Luque-Baena, R. M., Ortiz-de-Lazcano-Lobato, J. M., & López-Rubio, E. (2021). Road pollution estimation from vehicle tracking in surveillance videos by deep convolutional neural networks. Applied Soft Computing, 113, 107950. doi:10.1016/j.asoc.2021.107950.

[53] Mądziel, M. (2023). Vehicle Emission Models and Traffic Simulators: A Review. Energies, 16(9), 3941. doi:10.3390/en16093941.

[54] Tsanakas, N., Ekström, J., & Olstam, J. (2017). Emission estimation based on cross-sectional traffic data. 22nd International Transportation and Air Pollution Conference, 15-16 November, 2017, Zürich, Switzerland.

[55] Guttikunda, S. (2008). Four Simple Equations for Vehicular Emissions Inventory. SIM-air Working Paper Series, 2.

[56] Abou-Senna, H., Radwan, E., Westerlund, K., & Cooper, C. D. (2013). Using a traffic simulation model (VISSIM) with an emissions model (MOVES) to predict emissions from vehicles on a limited-access highway. Journal of the Air & Waste Management Association, 63(7), 819-831. doi:10.1080/10962247.2013.795918.

[57] Randall, S., Sivertsen, B., Ahammad, S. S., Cruz, N. D., & Dam, V. T. (2015). Emissions Inventory for Dhaka and Chittagong of Pollutants PM10, PM2.5, NOx, SOx, and CO. Norwegian Institute for Air Research Dhaka (NILU), Report No. OR 45/2014.

[58] Farid, A., Hussain, F., Khan, K., Shahzad, M., Khan, U., & Mahmood, Z. (2023). A Fast and Accurate Real-Time Vehicle Detection Method Using Deep Learning for Unconstrained Environments. Applied Sciences (Switzerland), 13(5), 3059. doi:10.3390/app13053059.

[59] Haque, A., Pulok, R. A., Rahman, M. M., Akter, S., Khan, N., & Haque, S. (2023). Recognition of Bangladeshi Sign Language (BdSL) Words using Deep Convolutional Neural Networks (DCNNs). Emerging Science Journal, 7(6), 2183–2201. doi:10.28991/ESJ-2023-07-06-019.

[60] Islam, M. A., Rahman, M. M., Shohan, A. A., Pulok, R. A., Akter, S., & Ahmed, M. T. (2023). Deep Neural Network-Based Approach to Identify the Paddy Leaf Disease using ResNet50-V2. 2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques, EASCT 2023, 1–6. doi:10.1109/EASCT59475.2023.10393586.

[61] Talaat, F. M., & ZainEldin, H. (2023). An improved fire detection approach based on YOLO-v8 for smart cities. Neural Computing and Applications, 35(28), 20939–20954. doi:10.1007/s00521-023-08809-1.

[62] Zhao, Z., Han, J., & Song, L. (2021). YOLO-Highway: An Improved Highway Center Marking Detection Model for Unmanned Aerial Vehicle Autonomous Flight. Mathematical Problems in Engineering, 1205153, 1–14. doi:10.1155/2021/1205153.