Sri Lanka Society of Transport and Logistics

Journal of South Asian Logistics and Transport
Volume 6 Issue 1 March 2026
JSALT 6.1.2: pp. 23-40
DOI: https://doi.org/10.4038/jsalt.v6i1.139

REAL-TIME VEHICLE CLASSIFICATION AND DIRECTIONAL COUNTING AT URBAN ROADS USING YOLOV11 UNDER HETEROGENEOUS TRAFFIC CONDITIONS

H M A G H C Abeyrathne[*] and T Sivakumar

Department of Transport Management and Logistics Engineering,
University of Moratuwa, Sri Lanka

[*] Correspondence should be addressed to *abeyrathnahmaghc.20@uom.lk*

ABSTRACT

Urban roads in developing countries face highly heterogeneous traffic, making manual data collection labour-intensive and error-prone. This study presents a deep learning-based approach for real-time vehicle classification and directional counting using the YOLOv11 framework. A custom dataset of 5,200 locally collected images, encompassing 11 vehicle classes defined by the Road Development Authority of Sri Lanka, was used to train two model variants: YOLOv11 medium and YOLOv11 small. To enable directional flow estimation, the system implements a line-pass-based tracking algorithm that distinguishes between up- and down-traffic streams. The system demonstrated high accuracy for common classes, such as cars (Upstream 98.8% and Downstream 96.8%), motorcycles (Upstream 88.8% and Downstream 97.6%), and three-wheelers (Upstream 85.8% and Downstream 98.4%). However, performance variations in other categories highlight the challenges of occlusion and visual ambiguity in mixed-traffic environments. The findings demonstrate that YOLOv11 models trained on locally representative datasets provide a reliable, scalable, and cost-effective alternative to manual survey methods. This approach provides a disaggregated data source to support traffic signal optimisation, intersection analysis, and urban transport planning under complex heterogeneous traffic conditions.

Keywords: *Real-time traffic monitoring, classified vehicle count, directional counting, YOLOv11, urban roads, heterogeneous traffic*

1. INTRODUCTION

Urban roads in developing countries experience highly heterogeneous traffic conditions, making the collection of accurate real-time traffic data a persistent challenge for effective traffic management and transport planning.

Effective traffic management heavily depends on the availability of accurate, continuous, and disaggregated traffic data, including vehicle volumes, directional flows, peak and off-peak variations, and vehicle composition. These data are essential for traffic signal optimisation, capacity analysis, infrastructure investment decisions, and broader urban transport planning.

In the absence of reliable real-time traffic information, traffic control strategies often rely on outdated or incomplete data, resulting in suboptimal signal timing, inefficient traffic management, and increased traffic congestion. Without real-time data on vehicle movements and queue lengths, traffic signal systems are constrained in their ability to implement the dynamic adjustments necessary for efficient traffic flow management. As a result of poor data-driven decision-making, many inefficiencies persist, mostly in cities that are developing commercial centres in Sri Lanka. Traffic data collection on roads is predominantly conducted through manual vehicle counting. Although manual surveys have long been considered the conventional baseline for traffic studies, they are labour-intensive, costly, and prone to human error. Invigilators at the site may experience fatigue, struggle to distinguish between vehicle classes in mixed traffic, or encounter adverse environmental conditions, such as poor lighting and weather. These limitations often result in inaccurate data, which can lead to ineffective policy decisions and misallocation of transport investments [1].

Furthermore, large-scale or continuous manual surveys require substantial human resources and repeated training, making them unsustainable for long-term traffic monitoring. The lack of systematic error-checking frameworks further reduces confidence in manually collected data, discouraging their use in evidence-based transport planning. Without adopting modern technology, such as deep learning-powered cameras for traffic monitoring, it is difficult to make systems more accurate and cost-effective [2].

Automated traffic data collection methods based on computer vision and deep learning have gained increasing global attention. Traffic data collection approaches can be broadly categorised into manual and automated methods [1]. In the manual method, traffic data are obtained through human intervention, whereas automated methods use deep learning techniques in computer vision. When the manual method can classify a limited number of vehicles, deep learning can classify all vehicle classes simultaneously using a single camera [3]. Consequently, deep learning-based Real-time Classified Vehicle Counting (RCVC) systems have emerged as promising alternatives to conventional traffic surveys. To address the inefficiencies in traditional traffic monitoring, this study employs deep learning models for real-time class- and vehicle-specific vehicle counting. Most available models are trained on datasets collected under homogeneous traffic conditions in developed countries, where vehicle types, lane discipline, vehicle spacing, and driving behaviour differ from those observed in Sri Lanka. As a result, these models often exhibit reduced accuracy when applied to local conditions characterised by mixed traffic, diverse vehicle types, and frequent stoppage. This highlights the need for locally adapted RCVC systems trained on datasets that reflect the heterogeneous traffic characteristics of Sri Lanka.

To address the above gap, it is necessary to investigate the application of advanced deep learning frameworks, such as the 'You Only Look Once' (YOLO) architecture, for real-time classified vehicle counting (RCVC) on urban roads. In the context of Sri Lankan transport planning, classification into 11 vehicle classes, as defined by the Road Development Authority (RDA), is essential for producing data that are directly applicable to national traffic analysis. Utilising localised vehicle datasets enables a better understanding of local heterogeneous traffic characteristics that generalised global models often overlook. Evaluating the effectiveness of such high-performance object detection models with a localised dataset is crucial for testing their potential as a reliable, scalable, and cost-effective alternative to manual traffic surveys in local contexts, ultimately providing a more robust foundation for evidence-based urban traffic management and infrastructure planning.

2. LITERATURE REVIEW

2.1. Importance of RCVC Systems

Accurate real-time traffic monitoring can help manage urban traffic congestion. The emergence of automated monitoring frameworks, specifically those that utilise deep learning and computer vision architectures, has enabled the classification of a greater range of vehicle types. This technological advancement improves the level of detail in analysing traffic data and allows for tailored changes to policies [2].

During traffic management studies, through movements at intersections [4] and mid-block counts (often characterised as straight-line bidirectional roads [5]) are also gaining importance, which is necessary for scrutinising complex traffic patterns to resolve congestion and enhance safety [4].

Traffic data are traditionally collected manually, which is both expensive and often prone to human error. To overcome these problems, traffic movements are observed using video cameras, and deep learning models systematically trained to classify vehicles from video clips are used in laboratories to count them by class [1]. These models offer remote accessibility, enabling scalable deployment in diverse environments [2].

The existing literature emphasises a research gap in optimising model training to achieve reliable real-time classification and counting of multiple vehicle types under heterogeneous traffic conditions on roads in developing countries [1], [3], [6].

2.2. Existing Method of RCVC System

Deep learning is a subset of machine learning. Various algorithms serve distinct functions in deep learning and are designated for specific responsibilities. Convolutional Neural Networks (CNNs) have emerged as a foundational technology in deep learning, drawing inspiration from the biological structure of human visual perception, unlike traditional machine learning methods, which require humans to manually identify and select important features from data. This process is prone to errors. CNNs use a two-stage architecture that combines a feature extractor with a classifier. This structure allows the model to automatically learn and recognise critical patterns, such as textures and edges, directly from raw input with minimal pre-processing. Because they can perform feature learning and classification simultaneously, CNNs are highly efficient and well-suited for real-time tasks such as object detection and image classification [7].

Building on the foundational role of CNNs, recent empirical studies have highlighted their continued dominance and versatility in computer vision tasks, such as image classification, object detection, and pixel-level segmentation. CNNs remain the preferred choice for designing lightweight models because of their superior generalisation capability, which is the ability of a model to perform accurately on new, unseen data. A key feature of its efficiency is local modelling, which uses lightweight depth-wise convolutions (a specialised layer that processes each image channel independently to reduce computational load) to achieve high performance with minimal processing requirements. Furthermore, adopting a multistage design framework that progressively reduces the image resolution across different network levels consistently improves the accuracy and efficiency of various vision models. These characteristics ensure that CNNs remain essential for real-time applications in which a balance between high speed and accuracy is critical [7].

2.3. YOLO Algorithm

The YOLO algorithm is considered a premier solution because it works quickly and accurately, achieving high precision in real-time operations[5]. The accuracy of the results from YOLO version 8 (YOLOv8) in vehicle-related global studies was outstanding, with scores of 96.58%, 96.325%, and 97.54% for detection, classification, and counting, respectively [8]. Building on the legacy, the YOLOv11 model has achieved significant improvements, as demonstrated in Figure 1, which presents a detailed performance comparison with YOLOv8 and YOLOv10. YOLOv11 shows noticeable enhancements in precision, recall, and mAP, mainly because later versions can detect motorcycles and bicycles too [9].

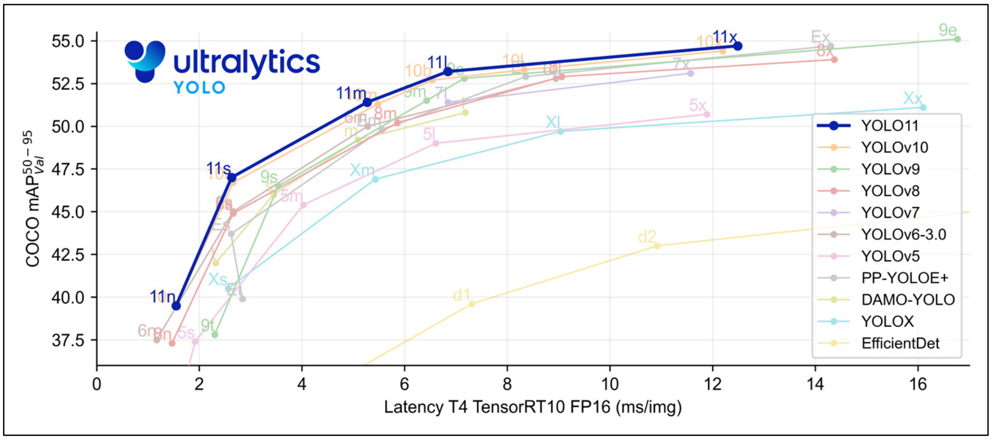


Figure 1: Performance of YOLO versions [10]

The cross-stage partial with spatial attention (CSPSA) block in YOLOv11 is an important architectural enhancement that enables more accurate object processing and detection while remaining resource-efficient. With the new features, the system can handle real-time workloads under all traffic conditions, even in inclement weather. YOLO11 processes images extremely quickly and accurately, enabling autonomous driving decisions without delay. Although YOLOv11 offers significant improvements in accuracy and functionality, it is designed to optimise computational resources for real-time applications, enabling scalability across multiple systems [9].

3. METHODOLOGY

Figure 2 illustrates the overall study methodology: (a) data collection, (b) data preparation, (c) model training, (d) counting, and (e) model evaluation.

3.1. Data Collection

Vehicle images and videos are required for this model. A locally collected dataset was used for training, validation, and testing. The eleven (11) vehicle classes proposed by the RDA, listed in Table 1, were considered in this study.

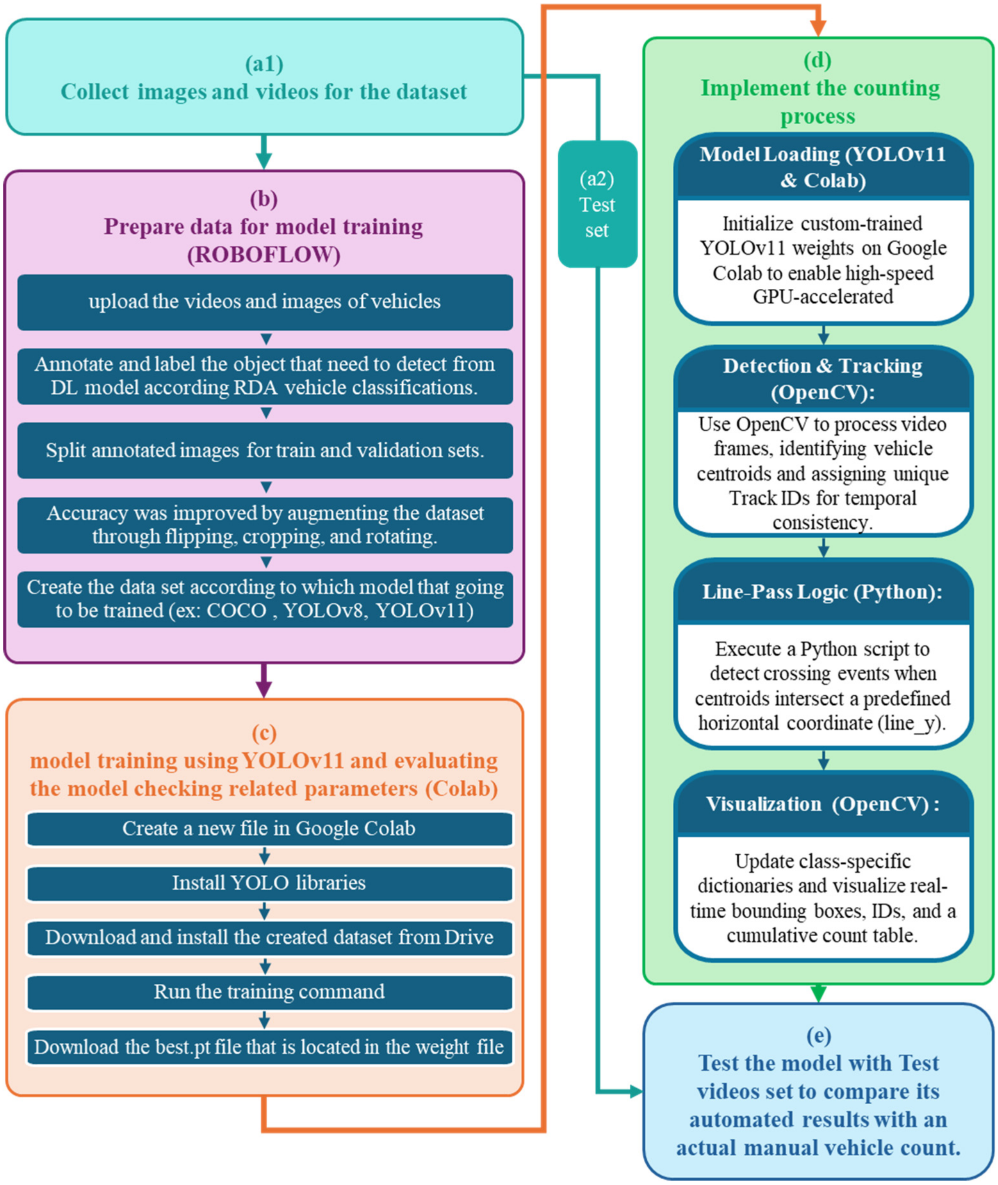


Figure 2: Model Development flow chart

Data were collected from multiple sources, including images and videos of vehicles, to develop a comprehensive dataset. To ensure that the model effectively learns the distinctive features of each class, an adequate number of vehicle samples for each category was collected. For training and validation, all images were combined into a single dataset (Figure 2(a1)), which was subsequently split after annotation. Simultaneously, the test data were collected separately as video footage and reserved exclusively for final model testing (Figure 2(a2)).

Table 1: RDA-defined vehicle classes

No	Vehicle Classification	No	Vehicle Classification
1	Motorcycle	7	Heavy goods vehicle
2	Three-Wheeler	8	Heavy goods vehicle with multiple axles
3	Car/Jeep/ 4x4 Double cab	9	Medium bus
4	Passenger van / Dual-purpose van	10	Large bus
5	Light goods vehicle	11	Tractor
6	Medium goods vehicle		

3.1.1. The dataset for model training and validation

The dataset was compiled from various image sources, as listed in Table 2, along with recorded videos, public datasets, and images from online platforms to ensure comprehensive coverage of the local vehicle categories. All videos were first converted to individual frames at their respective frame rates, yielding a total of 5,318 images for the dataset.

Table 2: Image data sources

Sources of Image	No of Images
Images were created from videos that were already collected in the past, according to the frame rate (converted video)	724
Google downloads, Facebook Marketplace, and groups. (Vehicle sale posts - Vehicle images were downloaded from Facebook sale posts because sellers typically capture multiple angles of their vehicles for buyers, providing diverse and detailed views for the dataset.)	3800
Known vehicle videos are converted to a set of images at a given frame rate using a converter. (converted YouTube video)	368
Recorded videos using my own cameras. (camera installed on Panadura pedestrian overpass to capture recorded vehicle footage)	426

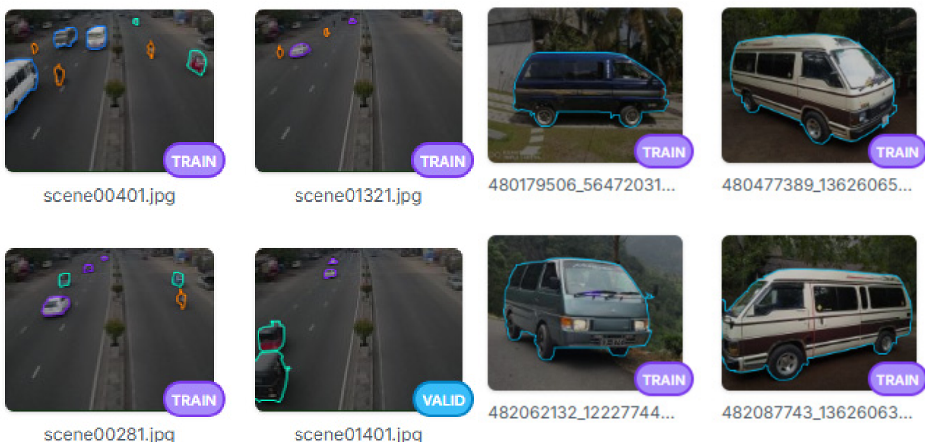
3.1.2. The dataset for model testing

The test dataset was curated exclusively for accuracy testing, ensuring complete isolation from the training and validation datasets. Video footage was captured at a road section in front of the Fort Railway station, with the camera strategically mounted at the centre of a pedestrian overpass to provide a bidirectional view of vehicular traffic (Figure 4). The data were recorded at a high-definition resolution of 1920 × 1080 pixels at 30 frames per second (fps).

3.2. Data Preparation

Before training, it is essential to organise the collected data in the correct format by annotating and labelling them. The Roboflow web platform was used for data annotation and labelling [11]. Data preparation using Roboflow proceeds through several stages, as illustrated in Figure 2(b). A project was created on the Roboflow platform, and the collected data were uploaded for training and validation purposes. One of Roboflow's built-in features automatically converts uploaded video footage into image frames at a user-defined rate. Eventually, 5,200 processed images were ready for annotation after removing a small number of duplicate images. The annotation process was conducted using Roboflow’s AI-assisted subject edge selection feature, which significantly reduced the manual effort and time. In total, more than 8,600 annotations were applied across the dataset, covering all the predefined vehicle classes to support effective model training and validation.

The prepared dataset was divided into 85% and 15% for training and validation using Roboflow to maximise the amount of data available for model learning while retaining a sufficient portion for performance evaluation and overfitting prevention. The training set was used to optimise the model parameters, whereas the validation set supported hyperparameter tuning and monitored convergence during training. This approach aligns with the standard practices in deep learning, ensuring robust model development and reliable accuracy assessment. The complete training and validation dataset can be accessed via the link below, including 4,458 training and 784 validation data points for classification into 11 vehicle classes. At the same time, Figure 3 shows a sample of (a) Traffic images and (b) Individual vehicles. *https://universe.roboflow.com/hasithproject/vehicle-classification-yjpih/dataset/3*



(a) Traffic images (b) Individual Images

Figure 3: Training and validation dataset

Data augmentation is a vital process in deep learning to increase the data size and improve the generalisation capability of the model, where the fed data are transformed by horizontal flipping, cropping, and ±15° rotation to generate an additional dataset [6],[3]. Creating such variations in the existing data helps prevent overfitting and improves precision, allowing models to capture the essence of unseen data versions [5],[8]. After augmentation, the total dataset of 13,000 images was converted to the YOLOv11 format as the final step in the data preparation.

3.3. Model Training

Based on the literature review and comparative analysis, Convolutional Neural Networks (CNNs) were identified as the most effective architecture for real-time vehicle detection in congested, mixed-traffic roads, like those in Sri Lanka [7], [12]

The YOLO framework was selected for its single-shot detection paradigm, which unifies object localisation (the process of identifying and determining the specific spatial position and boundaries of an object within an image [13]) and classification into a single regression task. This regression-based approach treats detection as a unified numerical problem and predicts the bounding box coordinates and class probabilities directly from the image pixels in a single forward pass [13].

YOLO architecture utilises a fixed grid system that partitions the input image into a symmetric matrix of cells (or meshes), where each cell predicts the properties of any object whose centre falls within its area. Every cell predicts bounding box coordinates, 'objectness scores' (a measure of the probability that a specific image region or window contains a distinct, recognisable object), and class probabilities simultaneously. By merging these steps, the architecture minimises computational redundancy and enables efficient end-to-end training and real-time inference, even under high-density traffic scenarios [13].

The anchor box mechanism of YOLO enhances detection robustness by predefining aspect ratio-specific priors (reference shapes), thereby improving the recognition of diverse vehicle geometries. Additionally, its multi-part loss function, which combines the localisation error, confidence loss, and classification loss (vehicle category probability), ensures balanced optimisation during training. Its Non-Maximum Suppression (NMS) algorithm ensures no duplicate detections, helping to eliminate false positives that often occur when a vehicle partially covers another vehicle as it manoeuvres through an intersection. These attributes make YOLO uniquely suited for heterogeneous traffic, where rapid inference and precise multi-class discrimination are paramount [13].

4. MODEL LEARNING PERFORMANCE

In accordance with the workflow shown in the flowchart in Figure 2(c), two YOLOv11 versions were trained on the created data set: the YOLOv11 medium (yolo11m) and YOLOv11 small (yolo11s) models. The YOLOv11m model has more parameters than the YOLOv11s model. Both models were trained on 1280 × 1280-pixel images, and training was performed for 180 epochs (an epoch represents one complete cycle through the entire training dataset during a model's learning process [14]) with a patience of 10 epochs. This indicates that when the mean Average Precision (mAP) is the same across consecutive 10 epochs, training is automatically stopped to eliminate overfitting when images are detected. Accordingly, the YOLOv11m training process was stopped after 159 epochs, and the YOLOv11s model was stopped at 64 epochs, as shown in Table 3. Model training was performed on Google Colab Pro, which provides high-performance Graphics Processing Units (NVIDIA A100) and sufficient memory to train the YOLOv11 models efficiently. Both models require over 10GB of Graphics Processing Unit (GPU) memory and more than 10GB of RAM during training. The model training procedure is shown in Figure 2(c).

Table 3: Model training last epochs

Epoch	GPU Mem	Box Loss	Cls Loss	DFL Loss	Instances	Size	P (Precision)	R (Recall)	mAP
62/100	10.8G	0.4476	0.3979	0.9935	18	1280	0.903	0.883	0.916
63/100	10.8G	0.4464	0.395	0.9915	42	1280	0.906	0.885	0.916
64/100	10.8G	0.4598	0.3898	0.9887	37	1280	0.906	0.885	0.916
(a) YOLOv11s									
157/180	16.3G	0.371	0.2971	0.9523	17	1280	0.909	0.891	0.925
158/180	16.4G	0.3709	0.298	0.9513	26	1280	0.908	0.891	0.925
159/180	16.3G	0.3735	0.2979	0.9525	19	1280	0.909	0.891	0.925
(b)YOLOv11m									

The training logs for the YOLOv11s (Table 3(a)) and YOLOv11m (Table 3(b)) models recorded the network's iterative convergence across multiple epochs, providing a quantitative assessment of model optimisation. The technical performance is categorised into loss components and validation metrics: box_loss represents the regression error in spatial localisation, cls_loss quantifies categorical misclassification, and dfl_loss evaluates the refinement of bounding box boundaries (Distribution Focal Loss). Model robustness was simultaneously evaluated using precision (P) and recall (R), along with the mean Average Precision (mAP).

Collectively, these metrics monitor the trade-off between computational efficiency, indicated by the GPU memory and inference speed, and the predictive accuracy of the respective architectures.

This study shows that the mAP remained unchanged during training, confirming that the model had converged (Table 3). In the last training rounds, YOLOv11m surpassed 92.6% mAP, whereas YOLOv11s achieved 91.6% mAP when evaluated on the validation images. Both models remained reliable for vehicle detection, were efficient to train, and performed well in terms of accuracy.

4.1. Vehicle Counting

As illustrated in Figure 2(d), the system was implemented in two primary phases: high-precision object detection, followed by motion-based counting logic. The process began by loading the custom-trained YOLOv11 weights into a Python-based environment integrated with OpenCV.

The first phase used YOLOv11 to identify vehicles in each frame. Once a vehicle is classified into one of the 11 predefined vehicle classes, the system assigns a unique Tracking ID. This "object persistence" feature is critical, as it ensures that the model maintains a continuous identity for each vehicle across consecutive frames, preventing a single vehicle from being registered as multiple detections during high-density traffic or partial occlusions.

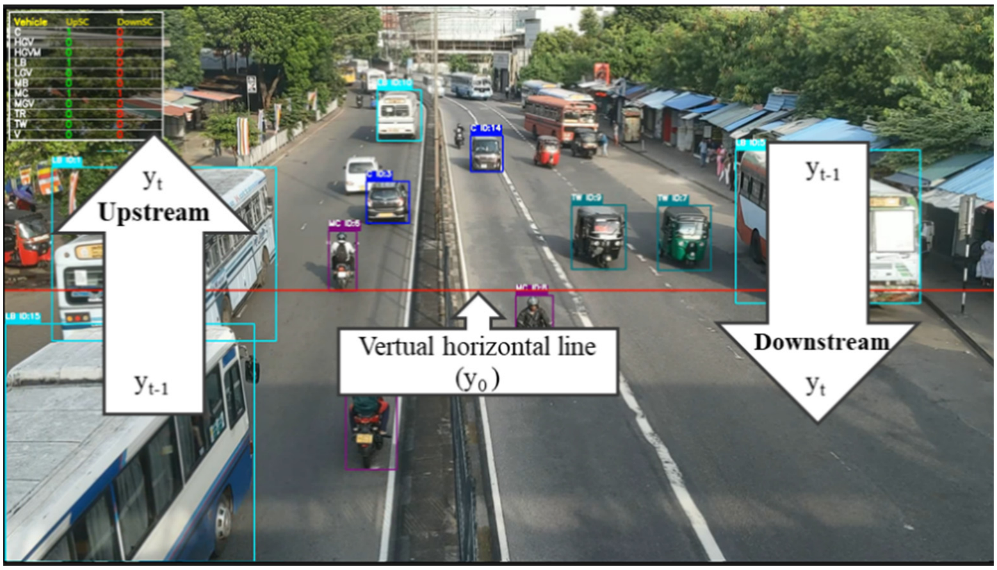


Figure 4: Downstream and upstream counts

A count was defined as a detected object crossing a virtual horizontal line across the entire width of the frame with a specific y-coordinate. This is called the "Line-Pass method" which functions as a virtual trigger. An algorithm monitors the centroid (geometric centre point (x, y)) of the vehicle's bounding box, and a spatial state change triggers the counting. The counting compares the centroid's position in the current bounding box(yt) against its position in the previous bounding box (yt-1) with the line's coordinate (y0). The system manages both Downstream Count (DownSC => yt-1 > y0 > yt) and Upstream Count (UpSC => yt-1 < y0 < yt) simultaneously, as shown in Figure 4.

The system generates an enhanced visual output by overlaying real-time analytical metadata on video footage, facilitating a comprehensive assessment of traffic behaviour and potential anomalies. To optimise clarity for the viewer, each of the 11 vehicle classes was assigned a distinct colour-coded bounding box with increased stroke thickness for better visibility (Figure 5). Class-specific labels were dynamically positioned at the top of each bounding box to provide immediate identification. To further enhance data visualisation, the cumulative traffic counts were displayed at the top-left corner of the video frame, providing a real-time summary of vehicle flow by category and direction (Figure 6). The integration of deep learning-based classified counting demonstrates a robust framework for automated traffic monitoring. Such high-fidelity visual data are instrumental for urban planning and the Development of Intelligent Transportation Systems (ITS), where precise classification and spatial awareness are critical.

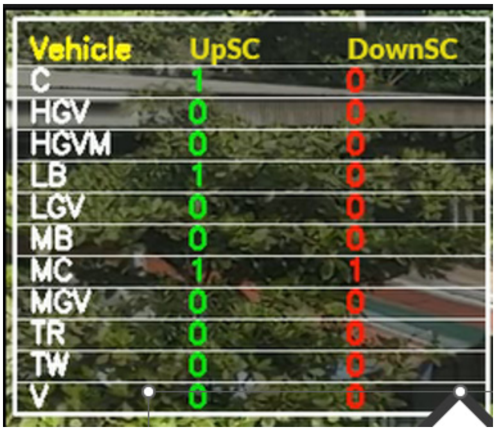


Figure 6: Vehicle counting user interface

Figure 5: Bounding box label and colours

4.2. Model Testing

As illustrated in the development flowchart in Figure 2(e), the final phase of the methodology involves a rigorous assessment of the validated RCVC system using real-world data comprising video sequences reserved exclusively for this stage to ensure unbiased results.

To establish a highly accurate baseline for comparison, manual vehicle counts were conducted on five-minute video segments. These segments were audited multiple times by human observers to eliminate errors and ensure that the ground truth (actual count) reached 100% accuracy. Shorter durations were used to mitigate human fatigue during the manual verification process. One of the primary advantages of a high-performance model is its ability to maintain reliable classification and counting accuracy over extended monitoring periods. Finally, by comparing the automated model output with the verified actual count, this study quantifies the model's precision and its ability to maintain tracking consistency across diverse traffic densities.

5. DATA ANALYSIS RESULTS

This section presents the results of analysing the test video datasets using the trained YOLOv11 models. The performance of the models was evaluated by comparing the model counts with the actual counts across multiple vehicle classes and travel directions using standard accuracy measurement methods. The analysis further examined class-wise and directional variations in accuracy to assess the effectiveness and reliability of the proposed approach under heterogeneous traffic conditions.

5.1. Accuracy Measurement

The objective of this study was to achieve automated vehicle counts (model count) that closely match the ground truth (actual count). The deviation between the model and actual counts can occur in multiple ways, such as the underestimation or overestimation of vehicle counts. For instance, when a vehicle is misclassified, the predicted count for its true class is reduced, resulting in a negative error. In contrast, the same vehicle may be incorrectly added to another class, resulting in a positive error. Additionally, counting inaccuracies may arise from duplicate counting of the same vehicle or missed detections at the defined counting line. To systematically assess these deviations, this study employed a standard accuracy evaluation metric for error measurement. Equation (1) provides the accuracy of vehicle classification by comparing the model count with the actual count.

$$\text{Accuracy \%} = \left(1 - \frac{|\text{Actual Count} - \text{Model Count}|}{\text{Actual Count}}\right) \times 100 \text{ ------------------- } (1)$$

where,

- Actual Count = Ground truth – Actual classified vehicle count
- Model Count = Vehicle count predicted by the model

5.2. Model Accuracy

Figure 7 presents the accuracy percentages for each vehicle class in both the upstream and downstream directions, as evaluated using the YOLOv11m and YOLOv11s models.

These results were obtained by comparing the ModelCount from the trained models with the ActualCount from the testing data (video datasets). The evaluation was conducted across 11 vehicle classes, some of which occurred frequently in the traffic flow and others less frequently. The model performance was assessed using the accuracy percentage metric, enabling a comparative analysis of the class-wise and directional (UpSC and DownSC) counting performance between the two YOLOv11 model variants. Each case was evaluated by comparing the model output to the ground truth.

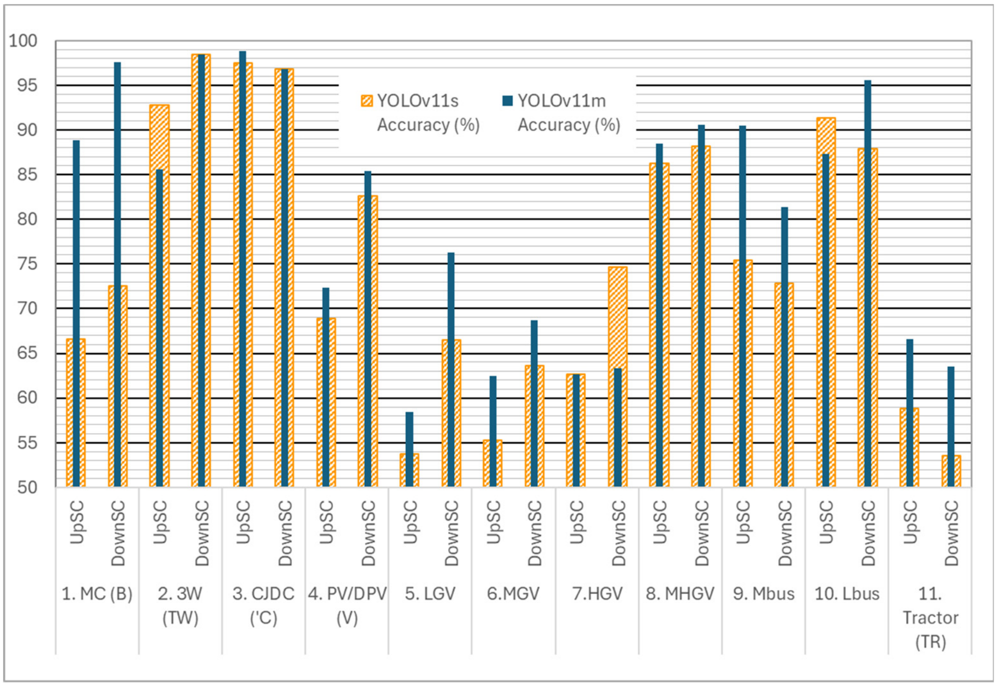


Figure 7: YOLO models and directions vs model Accuracy %

5.2.1. Overall performance

Overall, the model accuracy was above 50% in all scenarios, with some exceeding 95% and the rest falling within that range. In most cases, the YOLOv11m model consistently outperformed YOLOv11s in both traffic directions. Furthermore, higher accuracy was consistently observed in the downstream direction (where the front views of vehicles were observed) than in the upstream direction (where the rear views of vehicles were observed), demonstrating that YOLOv11m's complex structure enables it to detect features in the frontal area more effectively.

5.2.2. Class-specific insights

The analysis showed that diverse types of classes were detected with varying efficiencies. Three-Wheels, Car/Jeep/4×4 double cabs, and large buses are the most common vehicles on Sri Lankan roads and are estimated with high accuracy (above 95%). Conversely, light goods vehicles (53.7% and 76.3%) and tractors (53.5% and 66.6%) exhibited lower classification accuracies, likely because of the visual ambiguity caused by heterogeneous cargo loads and insufficient, highly variable training data, particularly regarding vehicle orientation. For instance, YOLOv11 accuracy for motorcycles dropped by 8.8% in upstream scenarios (97.6% downstream vs. 88.8% upstream), likely due to the reduced prominence of rear features, such as taillights.

5.2.3. Model-specific insights

Although YOLOv11m generally demonstrated superior performance, it struggled with Medium Goods Vehicle prediction, potentially because of visual similarities with other goods-moving vehicle categories. There are four main types of trucks: light goods vehicles, medium goods vehicles, heavy goods vehicles, and multi-axle heavy goods vehicles. YOLOv11s outperformed YOLOv11m in selected cases, such as three-wheelers (98.4%) in UpSC and heavy goods vehicles (74.6%) in DownSC, respectively. A comparison between the two models revealed that the motorcycle category exhibited the largest accuracy gap, amounting to 22.2% for upstream traffic and 25.1% for downstream traffic.

6. DISCUSSION AND IMPLICATIONS

This study demonstrates that YOLOv11-based real-time vehicle classification systems can effectively monitor heterogeneous urban traffic when trained using locally representative data. YOLOv11 m's superior performance highlights the importance of model capacity for capturing fine-grained visual features, especially for approaching vehicles. Directional and class-wise accuracy variations revealed challenges with obstructions, rear-view perspectives, and underrepresented vehicle types. Despite these limitations, the system achieved sufficient accuracy for operational traffic analysis, making it a viable alternative to manual surveys. Camera-based RCVC solutions thus offer continuous, cost-efficient, and disaggregated traffic data, enabling evidence-based signal optimisation, intersection assessment, and sustainable transport planning in resource-constrained urban environments.

To address the identified limitations and enhance the accuracy of the YOLOv11 vehicle classification, several future improvements are proposed. Increasing the size and diversity of the training and validation datasets is expected to strengthen the model's ability to generalise across different traffic conditions and vehicle appearances. Targeted data augmentation strategies should be applied to vehicle classes that exhibit lower accuracy, thereby reducing class imbalance and mitigating biases arising from limited sample representation. Despite the promising performance of the proposed system, certain constraints inherent to video-based traffic analysis remain, including sensitivity to the camera viewpoint, occlusions, and varying illumination conditions. These factors can affect visibility and classification reliability, especially under low-light or suboptimal camera placement scenarios. Addressing these challenges through improved data diversity and environmental robustness is essential for further improving system performance in real-world traffic-monitoring applications.

7. CONCLUSION

This study demonstrated that the YOLOv11m-based Real-Time Classified Vehicle Counting (RCVC) system provides a robust and scalable alternative to traditional manual traffic surveys in heterogeneous urban environments. By demonstrating significant precision (more than 90% accuracy) across standard vehicle classes, this study proves that computer vision models can bridge the critical data gap in regions where sophisticated sensor infrastructure is absent [15].

YOLOv11m model offers superior detail detection due to its architectural complexity. These findings provide a cost-effective framework for urban transport planning, enabling data-driven decisions for signal optimisation and traffic management in developing regions. The practical implementation revealed that both tested variants performed effectively. Ultimately, this study provides a cost-efficient framework for transport authorities to modernise traffic monitoring, supporting sustainable urban mobility through data-driven decision-making rather than outdated estimates. By integrating these automated solutions, cities can achieve more responsive and intelligent transportation.

REFERENCES

[1] R. A. P. U. S. Perera, H. M. O. K. Herath, B. Ayesha, T. Sivakumar, A. S. Kumarage, and A. S. Perera F, 'Evaluation of Manual and Video-Based Automated Classified Vehicle Counting Methods for Heterogeneous Traffic Flow', 2021. Accessed: Jan. 05, 2026. [Online]. Available: https://easts.info/on-line/proceedings/vol.13/pdf/PP3016_R1_F.pdf

[2] P. Premaratne, I. Jawad Kadhim, R. Blacklidge, and M. Lee, 'Comprehensive review on vehicle Detection, classification, and counting on highways', *Neurocomputing*, vol. 556, p. 126627, Nov. 2023, doi: 10.1016/J.NEUCOM.2023.126627.

[3] N. Pethiyagoda *et al.*, 'Deep Learning-Based Vehicle Type Detection and Classification', *International Journal of Computational and Applied Mathematics & Computer Science*, vol. 3, pp. 18–26, Jun. 2023, doi: 10.37394/232028.2023.3.3.

[4] A. Vikram, J. Akshya, S. Ahmad, L. Jerlin Rubini, S. Kadry, and J. Kim, 'Deep Learning Based Vehicle Detection and Counting System for Intelligent Transportation', *Computer Systems Science and Engineering*, vol. 48, no. 1, p. `115-130, 2024, doi: 10.32604/csse.2023.037928.

[5] J. Da Wu, B. Y. Chen, W. J. Shyr, and F. Y. Shih, 'Vehicle classification and counting system using yolo object detection technology', *Traitement du Signal*, vol. 38, no. 4, pp. 1087–1093, Aug. 2021, doi: 10.18280/ts.380419.

[6] A. Vajeeran and G. De Silva, 'Identification of Effective Intersection Control Strategies During Peak Hours', *Transportation Research Procedia*, vol. 48, pp. 687–697, Jan. 2020, doi: 10.1016/J.TRPRO.2020.08.069.

[7] F. M. Shiri, T. Perumal, N. Mustapha, and R. Mohamed, 'A Comprehensive Overview and Comparative Analysis on Deep Learning Models: CNN, RNN, LSTM, GRU', *Journal on Artificial Intelligence*, vol. 6, no. 1, pp. 301–360, Mar. 2025, doi: 10.32604/jai.2024.054314.

[8] S. B. Neamah and A. A. Karim, 'Real-time Traffic Monitoring System Based on Deep Learning and YOLOv8', *ARO-THE SCIENTIFIC JOURNAL OF KOYA UNIVERSITY*, vol. 11, no. 2, pp. 137–150, Nov. 2023, doi: 10.14500/aro.11327.

[9] M. Al and R. Alif, 'YOLOv11 for Vehicle Detection: Advancements, Performance, and Applications in Intelligent Transportation Systems', Oct. 2024, Accessed: Jan. 05, 2026. [Online]. Available: https://arxiv.org/pdf/2410.22898

[10] 'Ultralytics YOLO11 - Ultralytics YOLO Docs'. Accessed: Jan. 19, 2026. [Online]. Available: https://docs.ultralytics.com/models/yolo11/

[11] S. K. Shandilya, A. Srivastav, K. Yemets, A. Datta, and A. K. Nagar, 'YOLO-based segmented dataset for drone vs. bird detection for deep and machine learning algorithms', *Data Brief*, vol. 50, p. 109355, Oct. 2023, doi: 10.1016/J.DIB.2023.109355.

[12] Y. Zhao, G. Wang, C. Tang, C. Luo, W. Zeng, and Z.-J. Zha, 'A Battle of Network Structures: An Empirical Study of CNN, Transformer, and MLP', Aug. 2021, Accessed: Jan. 05, 2026. [Online]. Available: https://arxiv.org/pdf/2108.13002

[13] Y. Zhang, Z. Guo, J. Wu, Y. Tian, H. Tang, and X. Guo, 'Real-Time Vehicle Detection Based on Improved YOLO v5', *Sustainability 2022, Vol. 14, Page 12274*, vol. 14, no. 19, p. 12274, Sep. 2022, doi: 10.3390/SU141912274.

[14] X. Zhang *et al.*, 'Target Classification Method of Tactile Perception Data with Deep Learning', *Entropy 2021, Vol. 23, Page 1537*, vol. 23, no. 11, p. 1537, Nov. 2021, doi: 10.3390/E23111537.

[15] A. Vikram, J. Akshya, S. Ahmad, L. Jerlin Rubini, S. Kadry, and J. Kim, 'Deep Learning Based Vehicle Detection and Counting System for Intelligent Transportation', *Computer Systems Science and Engineering*, vol. 48, no. 1, p. `115-130, 2024, doi: 10.32604/csse.2023.037928.