Journal on Artificial Intelligence

Tech Science Press

Doi:10.32604/jai.2025.069008

ARTICLE

Attention-Augmented YOLOv8 with Ghost Convolution for Real-Time Vehicle Detection in Intelligent Transportation Systems

Syed Sajid Ullah[1,*], Muhammad Zunair Zamir[2], Ahsan Ishfaq[2] and Salman Khan[1]

[1]School of Energy and Electrical Engineering, Chang'an University, Xi'an, 710064, China
[2]School of Information Engineering, Chang'an University, Xi'an, 710064, China

*Corresponding Author: Syed Sajid Ullah. Email: sajid@chd.edu.cn

Received: 11 June 2025; Accepted: 24 July 2025; Published: 29 August 2025

ABSTRACT: Accurate vehicle detection is essential for autonomous driving, traffic monitoring, and intelligent transportation systems. This paper presents an enhanced YOLOv8n model that incorporates the Ghost Module, Convolutional Block Attention Module (CBAM), and Deformable Convolutional Networks v2 (DCNv2). The Ghost Module streamlines feature generation to reduce redundancy, CBAM applies channel and spatial attention to improve feature focus, and DCNv2 enables adaptability to geometric variations in vehicle shapes. These components work together to improve both accuracy and computational efficiency. Evaluated on the KITTI dataset, the proposed model achieves 95.4% mAP@0.5—an 8.97% gain over standard YOLOv8n—along with 96.2% precision, 93.7% recall, and a 94.93% F1-score. Comparative analysis with seven state-of-the-art detectors demonstrates consistent superiority in key performance metrics. An ablation study is also conducted to quantify the individual and combined contributions of Ghost Module, CBAM, and DCNv2, highlighting their effectiveness in improving detection performance. By addressing feature redundancy, attention refinement, and spatial adaptability, the proposed model offers a robust and scalable solution for vehicle detection across diverse traffic scenarios.

KEYWORDS: YOLOv8n; vehicle detection; deformable convolutional networks (DCNv2); ghost module; convolutional block attention module (CBAM); attention mechanisms

1 Introduction

Vehicle detection has become increasingly critical in various domains, including autonomous driving, traffic monitoring, and intelligent transportation systems (ITS). The ability to accurately identify vehicles in diverse and complex environments is essential for ensuring safety, optimizing traffic flow, and enhancing the overall efficiency of transportation networks. As urbanization accelerates and vehicle populations grow, the challenges associated with vehicle detection, such as occlusion, varying scales, and diverse orientations, have intensified. Consequently, there is a pressing need for advanced detection algorithms that can effectively address these challenges while maintaining computational efficiency [1].

Real-time and accurate vehicle detection remains fundamental to modern intelligent transportation systems. While deep learning-based object detection models have shown remarkable progress, they still face challenges in detecting small, occluded vehicles in complex urban scenarios. Additionally, computational efficiency remains a crucial factor for real-world deployment, particularly in edge devices and autonomous vehicles with limited resources.

Table 1: Evolution of YOLO models

Ref.	YOLO Version	Year	Main Advancements
[2]	YOLOv1	2015	Real-time detection using a grid-based unified model for detection and regression.
[3]	YOLOv2	2016	Added anchor boxes, dimension clustering, and multi-scale training for small object detection.
[4]	YOLOv3	2018	Introduced Darknet-53 backbone, multi-label classification, and a three-scale feature pyramid.
[5]	YOLOv4	2020	Incorporated Mosaic augmentation, CSPDarknet53 backbone, and PAN/SAM neck improvements.
[6]	YOLOv5	2020	PyTorch-native version with autoanchor, hyperparameter evolution, and simplified architecture.
[7]	**YOLOv8**	**2023**	Used dynamic convolution, an anchor-free split head, and trained on multiple datasets.

The YOLO (You Only Look Once) family of detectors has emerged as a leading framework for real-time object detection due to its balance of speed and accuracy. The evolution of YOLO models are demonstrated in Table 1. YOLOv8, the latest iteration, incorporates several advancements over its predecessors, including improved backbone networks and enhanced feature extraction mechanisms. However, despite its strengths, YOLOv8 still faces limitations in detecting small and occluded objects, which are prevalent in real-world scenarios [8]. While multiple variants of YOLOv8 exist, our work utilizes YOLOv8n due to its lightweight architecture and superior efficiency, making it suitable for deployment in resource-constrained environments.

This study aims to address these limitations by integrating three innovative components: the Ghost Module, the Convolutional Block Attention Module (CBAM), and Deformable Convolutional Networks v2 (DCNv2) [9–11]. The Ghost Module is designed to improve feature extraction by reducing redundancy in the feature maps, thereby allowing the model to focus on more relevant features [9]. CBAM further enhances this capability by applying attention mechanisms that enable the model to prioritize important features across multiple scales, which is particularly beneficial in complex environments where vehicles may be partially obscured or vary significantly in size [10]. DCNv2 introduces learnable offsets and modulation scalars into convolutional kernels, improving the model's robustness to geometric transformations in object appearance [11]. Additionally, the integration of DCNv2 increases the model's robustness to variations in object position, scale, and occlusion, which are common challenges in vehicle detection tasks [12].

The main contributions of this paper are:

- A novel YOLOv8-based architecture that strategically integrates Ghost Module, CBAM, and DCNv2 to enhance vehicle detection performance.
- A comprehensive evaluation of the proposed architecture on standard vehicle detection benchmarks, demonstrating significant improvements in accuracy and computational efficiency.
- An ablation study that quantifies the individual and combined contributions of Ghost Module, CBAM, and DCNv2, validating their impact on overall detection performance.
- Empirical evidence showing that the proposed model significantly outperforms the baseline YOLOv8.

The structure of this paper is as follows: Section 2 reviews the related work on vehicle detection methods. Section 3 presents the methodology, including the proposed YOLOv8-based model, the integration of Ghost Module, CBAM, and DCNv2, as well as the dataset and evaluation metrics. Section 4 presents a comparative analysis of the proposed model with baseline and state-of-the-art methods, followed by a discussion of the results, highlighting key improvements and their implications. Section 5 concludes the study and suggests future research directions.

2 Related Work

Vehicle detection has garnered significant attention in recent years, particularly with the advent of deep learning techniques that have revolutionized the field of computer vision. The YOLO family of models has established itself as a leading framework for real-time object detection, with each iteration introducing significant improvements. The evolution from YOLOv1 to YOLOv8 has been extensively documented [13], with initial versions introducing grid-based detection and anchor boxes, while later iterations like YOLOv4 and YOLOv5 brought enhanced performance through optimizations including advanced data augmentation techniques and more flexible architectures. YOLOv8, introduced in 2023, represents the current state-of-the-art, featuring significant architectural improvements, including a more efficient backbone, enhanced feature extraction, and better accuracy-speed trade-offs [14].

Numerous studies have explored various methodologies to enhance vehicle detection accuracy and efficiency. Yang et al. proposed an improved vehicle detection system that integrates double-layer Long Short-Term Memory (LSTM) modules, demonstrating enhanced performance on the KITTI dataset and a self-built dataset collected from Taiwanese highways [15]. In tunnel environments, Kim's research revealed the limitations of traditional vehicle detection methods, such as the Aggregate Channel Features (ACF) and Fast R-CNN, which struggled in conditions with reduced visibility [16]. Chavan's work on YOLOv5 illustrates the versatility of deep learning models in vehicle detection, emphasizing the importance of feature engineering and the sliding-window technique [17]. Moreover, Jiang's study on multi-target detection for roads utilized manually designed features combined with classifiers like SVM and AdaBoost [18], while Xu and Chen introduced a lightweight vehicle detection network based on YOLOv5 for intelligent transportation applications [19].

The integration of attention mechanisms has been a focal point in recent research for enhancing object detection performance. Lee et al. proposed a convolutional neural network that employs selective multi-stage feature fusion to enhance vehicle detection performance [20]. CBAM, proposed by [10], has emerged as a particularly effective attention mechanism that sequentially applies channel and spatial attention to refine feature maps. Several studies have demonstrated the effectiveness of CBAM in object detection tasks, with [21] reporting significant improvements in detection accuracy, particularly for small objects, when integrating CBAM into YOLOv5. Reference [22] combined CBAM with feature pyramid networks for enhanced object detection in aerial imagery. Furthermore, reference [23] addressed the challenge of occlusion in vehicle detection by incorporating countermeasure learning into the RCNN framework, thereby improving detection accuracy in cluttered environments.

The Ghost Module, introduced by [9], addresses the computational inefficiency of traditional convolutional networks by generating more feature maps with fewer parameters. This approach significantly reduces the computational cost and model size without substantial performance degradation by applying standard convolution to generate a small set of intrinsic feature maps, followed by cheap linear operations to generate more feature maps. Similarly, Deformable Convolutional Networks v2 (DCNv2), proposed by [24], enhance the original DCN by introducing a modulation mechanism that adjusts the influence of each sampling point. This enables more precise control over the deformation process and improved feature extraction for objects with complex shapes.

Table 2: Literature review: recent detection methods on KITTI

Reference	Method	Dataset (s)	Key insights
[25]	Pseudo-RGB BEV to YOLOv8	KITTI, TIAND	LiDAR to BEV-RGB conversion achieves 86.3% mAP with real-time inference at 28.3 FPS.
[26]	YRDM: Efficient mining + dynamic confidence	KITTI	15% reduction in parameters while maintaining 80% mAP; achieves 35 FPS on RTX-3090.
[27]	YOLOv8 + CBAM + multi-scale fusion	KITTI	Improves small-object recall by 4% over baseline YOLOv8 with 10% latency increase.
[14]	Motion-aware YOLOv8	KITTI, LASIESTA	Preprocessing improves moving-object precision by 6%; struggles with static targets in cluttered scenes.
[28]	YOLOv5s + pruning and quantization	KITTI	Model size reduced by 3× while maintaining 75% mAP (12% drop in low-light conditions).
[29]	GBForkDet: Ghost module + Bi-OD Neck	KITTI	Ghost modules reduce parameters by 20%; redesigned neck boosts mAP on occlusions by 5%.
[30]	Z-YOLOv8s: SSD-style head + WIoU loss	KITTI	SSD-style head increases mAP by 3% in dense urban scenes; highway performance not reported.

Recent advancements have also seen the emergence of hybrid approaches that combine different sensing modalities. Reference [31] proposed a real-time vehicle detection algorithm that fuses vision and LiDAR point cloud data, effectively leveraging the strengths of both modalities to enhance detection accuracy, particularly for small and occluded targets. Also, Table 2 is showing brief Summary of Related Works including Datasets Used, and Key Contributions. The landscape of vehicle detection research is characterized by a continuous pursuit of improved accuracy and efficiency through innovative methodologies, including deep learning architectures, attention mechanisms, and multi-sensor fusion techniques.

Despite the advancements in vehicle detection models, existing architectures often struggle with computational efficiency and robustness in complex environments. Traditional YOLO-based models, including the baseline YOLOv8, exhibit limitations in accurately detecting small or partially occluded vehicles due to insufficient feature extraction and poor adaptation to geometric variations. Additionally, standard convolutional modules in these models tend to generate redundant features, which not only increase computational cost but also reduce the model's efficiency. Moreover, the lack of attention mechanisms often leads to decreased performance in scenarios where vehicles vary significantly in size or are surrounded by cluttered backgrounds. These challenges limit the practical application of such models in real-time and dynamic settings.

To address these shortcomings, the proposed model strategically integrates the Ghost Module to reduce redundancy, CBAM to enhance feature discrimination through attention mechanisms, and DCNv2 to handle geometric transformations. These modifications collectively improve detection accuracy, robustness, and computational efficiency, making the proposed model more suitable for real-world intelligent transportation systems.

3 Methodology

3.1 YOLOv8 Architecture

The YOLOv8 architecture represents a state-of-the-art single-stage object detection framework characterized by its efficient and robust design. Building upon the strengths of previous YOLO versions, YOLOv8 introduces several novel improvements aimed at enhancing both performance and computational efficiency. The architecture is composed of three primary components: the Backbone Network (CSPDarknet), the Neck (Feature Pyramid Network), and the Detection Head.

3.1.1 Backbone Network (CSPDarknet)

The Backbone Network in YOLOv8 is built upon CSPDarknet, which utilizes Cross-Stage Partial (CSP) connections to enhance feature extraction while maintaining computational efficiency. The CSP connections split the feature maps into two parts and process them separately, thereby reducing computational cost without sacrificing feature quality. This approach is particularly beneficial for maintaining strong gradient flow during training, leading to faster convergence and better performance.

CSPDarknet incorporates a series of convolutional layers with varying kernel sizes, allowing the network to learn features at multiple scales. This design is crucial for detecting objects of different sizes, enabling the model to handle both small and large objects effectively. The architecture also integrates residual blocks and dilated convolutions, which help in capturing fine-grained details while ensuring minimal computational overhead. These elements make CSPDarknet both lightweight and powerful, ensuring robust feature extraction.

Additionally, YOLOv8 uses advanced activation functions such as Mish or SiLU. These activation functions have been shown to improve model convergence and performance, offering smoother gradients during backpropagation and better generalization. Batch normalization is applied after each convolutional operation, stabilizing the learning process and accelerating convergence by reducing internal covariate shift. This combination of features allows YOLOv8 to effectively learn and generalize complex patterns from the input data.

3.1.2 Neck (Feature Pyramid Network)

The Neck of YOLOv8 is designed to aggregate and refine features from multiple scales, making the model capable of detecting objects at various levels of abstraction. The Feature Pyramid Network (FPN) is used to combine low-level, high-resolution features with high-level, semantically rich features. This multi-scale feature fusion enables the model to improve detection performance, especially for objects that appear at different sizes within the image.

The network also employs the Path Aggregation Network (PAN), which enhances the feature fusion process by improving the flow of information across different stages. Unlike traditional FPNs, PAN introduces both top-down and bottom-up pathways, allowing the network to incorporate fine-grained low-level features alongside high-level context. This dual information flow ensures that the model is capable of detecting small objects in the foreground while maintaining the ability to recognize large objects in the background.

In addition to PAN, YOLOv8 employs cross-stage feature fusion and adaptive feature selection mechanisms to dynamically prioritize features based on the input data. These methods help the network to focus on the most relevant features for object detection tasks, leading to improved accuracy and efficiency. By leveraging multiple feature scales and refining them through adaptive fusion strategies, the Neck component ensures that the network can detect objects of varying sizes and complexities effectively.

3.1.3 Detection Head

The Detection Head in YOLOv8 is designed to generate accurate object predictions. One of the key innovations in YOLOv8 is its adoption of an anchor-free detection strategy. Unlike traditional object detection models, which rely on predefined anchor boxes, YOLOv8 directly predicts the bounding box coordinates relative to the grid cells. This eliminates the need for anchor box generation and matching, simplifying the detection process and making the model faster and more flexible.

The Detection Head is also decoupled into two distinct branches: one for classification and one for bounding box localization. This decoupling allows each task to be handled independently, reducing the competition between classification and localization tasks that often occurs in traditional models. As a result, the model can focus more on each task, leading to more accurate predictions. The decoupling also helps reduce the complexity of the model, allowing it to run more efficiently without sacrificing performance.

In addition to bounding box coordinates and class probabilities, the Detection Head also computes confidence scores, which indicate the model's certainty about its predictions. This is crucial for object detection, as it helps in filtering out low-confidence detections that are likely false positives. The combination of anchor-free detection and a decoupled head contributes significantly to the overall speed and accuracy of YOLOv8, making it suitable for a variety of real-time applications.

3.1.4 Computational Efficiency and Speed

YOLOv8 is designed to be computationally efficient while maintaining high accuracy, making it ideal for real-time applications. The use of CSP connections in the Backbone Network reduces the computational cost without compromising the quality of feature extraction. Additionally, the decoupled Detection Head and anchor-free detection strategy simplify the model's architecture, resulting in faster inference speeds. These design choices enable YOLOv8 to achieve high performance on a wide range of hardware, from GPUs to edge devices.

The network also employs several performance optimization techniques. Quantization and pruning are used to reduce the model's size and computational requirements, making it more suitable for deployment on resource-constrained devices. Quantization reduces the precision of the model's weights, while pruning removes redundant neurons and weights, resulting in a smaller and more efficient model. Furthermore, gradient checkpointing is employed to minimize memory usage during training, allowing the model to scale more effectively and be trained on deeper architectures.

Real-time inference is achieved through a combination of reduced floating-point operations (FLOPs), hardware acceleration, and optimized layer execution. These optimizations ensure that YOLOv8 can perform object detection tasks at high speed, even in demanding applications such as autonomous driving, video surveillance, and robotics. By optimizing both the architecture and the computational processes, YOLOv8 strikes a perfect balance between accuracy, speed, and resource efficiency.

3.1.5 Performance Optimization

In addition to the structural optimizations mentioned above, YOLOv8 includes several performance optimization strategies to further enhance its efficiency. Techniques like model quantization and pruning not only reduce the model size but also decrease the number of operations required for inference, leading to faster processing times and lower memory usage. These optimizations make YOLOv8 suitable for real-time deployment on devices with limited computational power, such as mobile phones, drones, and embedded systems.

The use of gradient checkpointing during training helps manage memory consumption by recomputing intermediate results instead of storing them, allowing for more efficient use of available resources. This is particularly important for training larger models that require substantial memory. These optimizations, combined with the efficient design of the YOLOv8 architecture, ensure that the model can achieve high detection accuracy while running efficiently in real-time environments.

Finally, YOLOv8 has been designed to perform well across various hardware platforms. The architecture is compatible with GPUs for high-speed training and inference, while the optimized model can also be deployed on edge devices such as mobile phones or embedded systems without significant loss in performance. This flexibility makes YOLOv8 a versatile choice for a wide range of object detection tasks, from autonomous vehicles to security systems.

3.2 Proposed Modifications

3.2.1 Ghost Module Integration

To reduce computational redundancy and model complexity, we integrate the Ghost Module [32] into the YOLOv8 architecture. Instead of generating all feature maps via standard convolution, Ghost Modules produce a small set of intrinsic feature maps followed by additional maps generated through cheap linear operations (e.g., depthwise convolutions). Mathematically:

$$X = W * I \tag{1}$$

$$Y_i = \Phi_i(X), \quad i = 1, 2, \ldots, s \tag{2}$$

$$Y = \mathrm{Concat}(X, Y_1, Y_2, \ldots, Y_s) \tag{3}$$

Here, I is the input, W is the convolution kernel, Φ_i represents linear transformations, and s is the number of ghost features per intrinsic map.

The Ghost Module is deployed in the neck of YOLOv8, replacing standard convolution blocks while preserving the network's feature hierarchy. This modification enhances feature diversity and extraction efficiency with minimal computational overhead, enabling real-time inference on resource-constrained devices. The convolutional layer and architecture of Ghost Module is shown in Figs. 1 and 2, respectively.

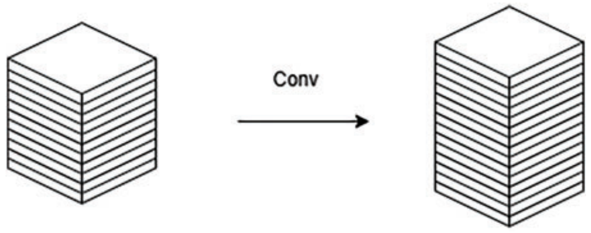


Figure 1: Convolution layer

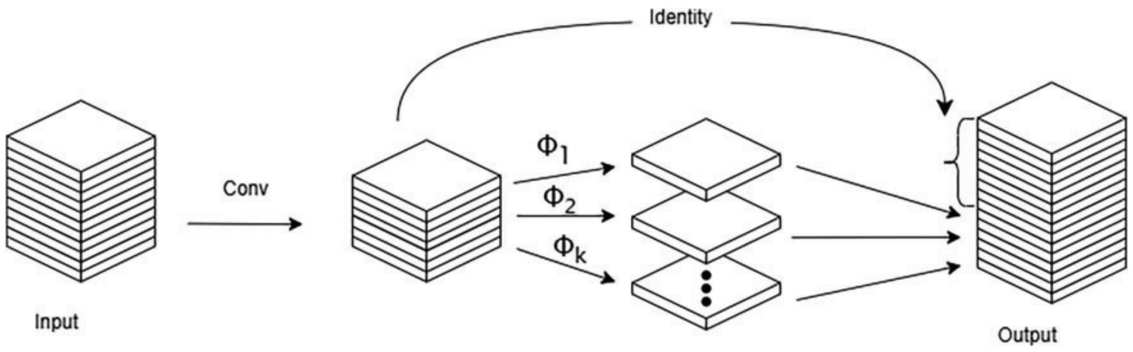


Figure 2: Ghost module

3.2.2 Convolutional Block Attention Module (CBAM)

The Convolutional Block Attention Module (CBAM) [33] enhances feature representations by applying attention mechanisms in both channel and spatial dimensions. It consists of two sequential submodules: the Channel Attention Module (CAM) and the Spatial Attention Module (SAM).

Channel Attention: Given an input feature map $F \in \mathbb{R}^{C\times H\times W}$, average-pooling and max-pooling are applied spatially:

$$F^c_{avg} = \text{AvgPool}(F), \quad F^c_{max} = \text{MaxPool}(F) \tag{4}$$

These are passed through a shared MLP with weights W_0, W_1:

$$M_c(F) = \sigma(\text{MLP}(F^c_{avg}) + \text{MLP}(F^c_{max})) \tag{5}$$

The refined feature is obtained by:

$$F' = M_c(F) \otimes F \tag{6}$$

Spatial Attention: From F', spatial descriptors are computed:

$$F^s_{avg} = \text{AvgPool}(F'), \quad F^s_{max} = \text{MaxPool}(F') \tag{7}$$

These are concatenated and passed through a convolution layer:

$$M_s(F') = \sigma(f^{7\times 7}([F^s_{avg}; F^s_{max}])) \tag{8}$$

The final output is:

$$F'' = M_s(F') \otimes F' \tag{9}$$

CBAM enhances the model's capacity to focus on informative features and suppress irrelevant background noise, leading to improved accuracy and robustness in tasks such as object detection, especially under occlusions or cluttered scenes.

3.2.3 Deformable Convolutional Networks v2 (DCNv2)

Deformable Convolutional Networks v2 (DCNv2) [24] extend the standard convolution operator by learning both spatial offsets and modulation scalars for each sampling location, thereby enabling adaptive reshaping of the receptive field to better capture geometric transformations and object deformations. In DCNv2, separate convolutional layers predict a set of offsets $\{\Delta p_k\}$ and modulation factors $\{\Delta m_k\}$, which are applied to the regular grid positions $\{p_k\}$ of the convolutional kernel. The output at location p_0 is computed as

$$y(p_0) = \sum_{k=1}^{K} w(k) \left[x\big(p_0 + p_k + \Delta p_k\big)\right] \Delta m_k, \tag{10}$$

where $w(k)$ denotes the weight of the k-th kernel element and $x(\cdot)$ denotes the input feature (with bilinear interpolation for non-integer locations). By jointly optimizing Δp_k and Δm_k along with the convolutional weights, DCNv2 dynamically adjusts both where and how much to sample, which substantially improves the modeling of objects under scale variation, occlusion, and non-rigid deformation. When integrated into the detection head of YOLOv8, these learned geometric adaptations result in more precise feature extraction and stronger localization performance, particularly in challenging scenarios involving irregular object shapes and poses.

3.2.4 Proposed Model Architecture

The proposed model builds upon the YOLOv8n architecture and introduces key architectural modifications to enhance detection performance while maintaining computational efficiency. This section details the structure and functionality of each component in the modified network, including the **Backbone**, **Neck**, and **Head**, along with the custom modules incorporated such as GhostConv, CBAM, and DCNv2. An overview of the proposed architecture is presented in Fig. 3.

The backbone of the model is designed to extract hierarchical visual features from the input image using a series of convolutional and residual connections. Initially, the input image is passed through multiple `CBS` blocks, each comprising a convolutional layer followed by batch normalization and the SiLU activation function. These layers enable early-stage feature extraction and downsampling.

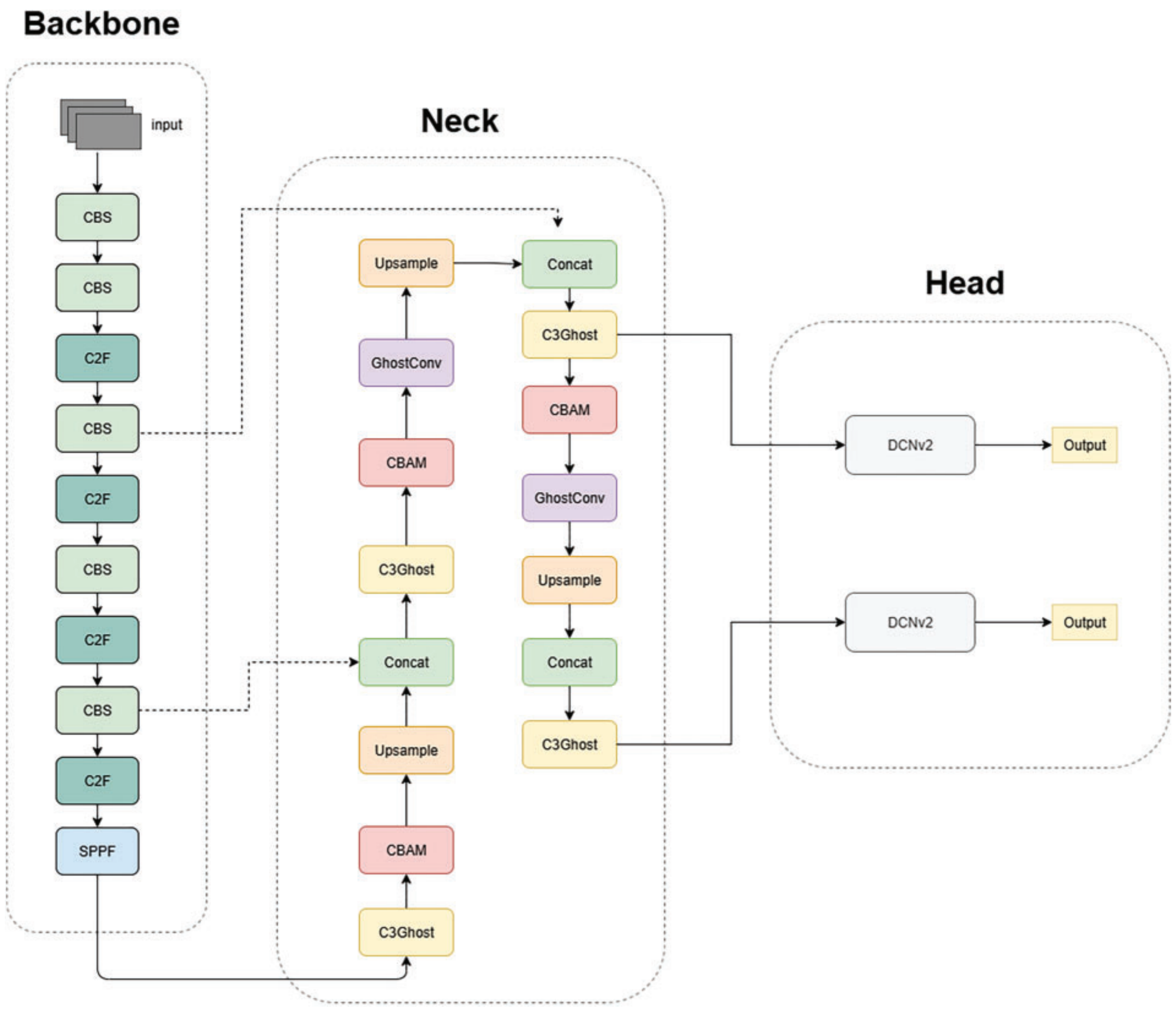


Figure 3: (Continued)

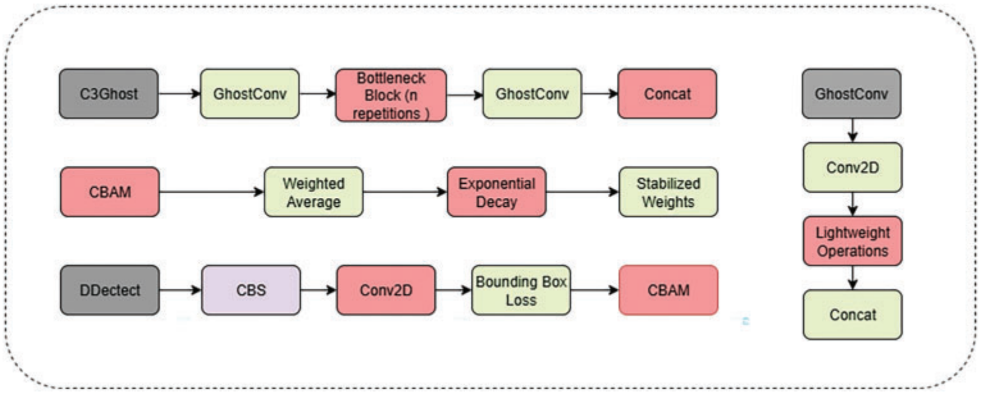


Figure 3: Architecture of the enhanced YOLOv8 framework integrating ghost modules, CBAM, and DCNv2

Subsequently, the model incorporates several `C2F` (Cross Stage Partial Fusion) modules, which are lightweight residual blocks that promote efficient feature reuse and better gradient flow across layers. This architecture is well-suited for deep but lightweight models, as it reduces the number of parameters while retaining expressive power.

To capture multi-scale spatial information, a Spatial Pyramid Pooling Fast (SPPF) module is used at the end of the backbone. This component aggregates features across different receptive fields, improving robustness to object scale variations, which is particularly beneficial in vehicle detection where size diversity is common.

The neck is responsible for aggregating features from different spatial resolutions and enhancing their discriminative capacity. In this design, the neck includes multiple `C3Ghost` modules—adapted from YOLO's C3 block but built with GhostConv layers for improved efficiency.

At each level of the neck, a `CBAM` (Convolutional Block Attention Module) is inserted to refine the extracted features by applying both channel-wise and spatial attention. This attention mechanism allows the model to selectively emphasize salient regions such as vehicle contours, lights, and structural parts, thus improving detection robustness in cluttered scenes.

The upsampling operations in the neck are followed by concatenation with feature maps from earlier stages (skip connections), enabling the integration of both low-level and high-level contextual information. These fused features are subsequently processed by GhostConv layers to retain lightweight processing while preserving informative representations.

The detection head is designed to generate final predictions for object location and classification. At each output scale, a `DCNv2` (Deformable Convolution v2) layer is employed. Unlike standard convolutions, DCNv2 dynamically adjusts the receptive field by learning spatial offsets, making it particularly effective at modeling geometric variations such as rotated or occluded vehicles.

Each DCNv2 output is passed through a 1×1 convolution layer to produce the final prediction tensor, which includes bounding box coordinates, object confidence scores, and class probabilities. The prediction tensor follows the format $[B, 3, H, W, 85]$, where 85 represents 4 box coordinates, 1 objectness score, and 80 class probabilities, consistent with the COCO detection standard. The overall training and inference procedure for the enhanced YOLOv8 model is outlined in Algorithm 1.

Algorithm 1: Enhanced YOLOv8 with Ghost, CBAM, and DCNv2

```
1: Input: Training images $I$, ground truth boxes $G$
2: Apply mosaic data augmentation on KITTI dataset
3: Integrate Ghost Module into YOLOv8 neck for efficient feature extraction
4: Insert CBAM in backbone and neck for attention-based refinement
5: Replace standard convolutions in head with DCNv2
6: for each epoch $t = 1$ to $T$ do
7:     for each batch of images do
8:         Extract multi-scale features $F$ using modified backbone
9:         Refine $F$ using CBAM and Ghost Module $\rightarrow F'$
10:        Predict bounding boxes $B$ using DCNv2 head
11:        for each predicted box $b \in B$ do
12:            Compute IoU between $b$ and ground truth $G$
13:            Calculate loss $L$ (classification, localization, IoU terms)
14:        end for
15:        Backpropagate loss and update model weights
16:    end for
17:    Evaluate precision, recall, and time on validation set
18: end for
19: Output: Trained YOLOv8-based detection model
```

The integration of GhostConv, CBAM, and DCNv2 contributes synergistically to the proposed model's performance. GhostConv significantly reduces the number of floating point operations (FLOPs) by generating feature maps through inexpensive transformations. CBAM modules help the model to concentrate on the most relevant spatial and channel-wise regions, thereby improving focus on key vehicle parts. DCNv2 further enhances spatial adaptability, enabling precise localization of vehicles under non-ideal conditions such as skewed angles and occlusions.

3.3 Dataset and Preprocessing

3.3.1 Data Composition

The dataset used for training and evaluating the proposed model is the KITTI dataset, a well-established benchmark in the field of object detection for autonomous driving. KITTI offers approximately 70,000 real-world images collected from diverse driving environments, including urban streets, highways, and rural roads. This variety enables robust model training by exposing it to different vehicle types, object scales, traffic densities, and scene complexities. Urban scenarios in the dataset often feature occlusions and dynamic traffic, while highway scenes provide more structured, high-speed conditions with fewer obstacles. The presence of vehicles at varying distances also allows the model to learn across multiple object scales, enhancing its generalization capability.

While KITTI is limited in terms of extreme lighting variations and adverse weather conditions, as it was mostly captured in clear daytime environments, it remains a strong foundation for evaluating models under realistic, everyday driving scenarios. The dataset includes several vehicle classes such as cars, vans, trucks, and trams, supporting a wide range of detection tasks relevant to intelligent transportation systems.

3.3.2 Preprocessing Techniques

The preprocessing pipeline applies several critical techniques to ensure that the data is properly prepared for training and that the model can generalize well to unseen data. Various steps are employed to improve the robustness and consistency of the input data.

Image Augmentation is used to artificially increase the size and diversity of the dataset. By applying transformations such as random horizontal flipping, color jittering, random cropping, and perspective distortions, the model is exposed to various visual changes in the input data. These augmentations help simulate different environmental conditions, like changes in viewing angle, lighting, and partial occlusions, ultimately making the model more resilient to real-world variability.

Normalization is employed to standardize the pixel values of the images, ensuring consistency across the dataset. The images are scaled to a predefined range, typically [0, 1], and then mean subtraction and standard deviation normalization are applied. These steps ensure that all images are on a similar scale and help the model converge more efficiently during training by mitigating issues like internal covariate shift.

Annotation Processing involves the standardization of bounding box coordinates to ensure they are consistently formatted across the dataset. Class labels are encoded into numerical representations, allowing the model to process the categorical data efficiently. Furthermore, care is taken to handle occlusions and truncations in the images, ensuring that partially visible objects are still accurately labeled and included in the model's training. This is crucial for tasks like vehicle detection, where occlusions are common in urban traffic and highway settings.

3.4 Evaluation Metrics

Model performance is evaluated using Precision, Recall, and mean Average Precision (mAP) [34]. Precision (positive predictive value) and Recall (true positive rate) are defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{11}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{12}$$

where TP, FP, and FN denote true positives, false positives, and false negatives, respectively. To aggregate detection accuracy across Q object classes, we define:

$$\text{mAP} = \frac{1}{Q}\sum_{q=1}^{Q} \text{AP}_q, \tag{13}$$

where AP_q is the average precision for class q computed over the precision–recall curve (e.g., by Riemann integration over IoU threshold).

3.5 Experimental Setup

The experiments were conducted on a high-performance computing environment to ensure efficient model training and evaluation. The detailed hardware and software configurations are presented in Table 3, while the key training hyperparameters such as batch size, optimizer, learning rate schedule, and input resolution are summarized in Table 4.

Table 3: Experimental setup for vehicle detection models

Component	Configuration details
Hardware	NVIDIA RTX 3090 GPU (24 GB GDDR6X), Intel Core i7-8750H CPU (6 cores @ 2.2 GHz), 16 GB DDR4 RAM, 1TB SSD storage
Software	PyTorch 1.10 framework with CUDA 11.3 acceleration, running in Python 3.8 environment with all necessary dependencies

Table 4: Training parameters for the proposed model

Component	Configuration details	
	Parameter	Value
Training parameters	Batch size	32
	Optimizer	Adam
	Initial learning rate	0.001
	Learning rate scheduler	Cosine annealing
	Early stopping	Patience = 10 epochs
	Input resolution	640 × 640 pixels
	Total epochs	150

4 Results and Discussion

4.1 Experimental Results

As illustrated in Fig. 4, the proposed improved YOLOv8 model demonstrates strong detection performance across a variety of real-world conditions captured from the KITTI dataset. The first set of images displays the original scenes, while the second set shows the corresponding detection results produced by the model. The model accurately identifies vehicles of different sizes, orientations, and levels of occlusion, which highlights its robustness and generalization capability. Even in scenarios with dense traffic, partial visibility, and variable lighting, the model maintains precise localization and classification. These qualitative results visually confirm the improvements achieved through the integration of the Ghost Module, CBAM, and DCNv2, which enhance feature representation, focus attention on relevant regions, and increase adaptability to spatial transformations, respectively.

Fig. 5 presents a comparative analysis of the proposed model's performance across four key evaluation metrics: precision, recall, F1-score, and mAP50. The Fig. 5 clearly illustrates the model's strong performance, with all metrics exceeding 93% and demonstrating consistent superiority. Precision reaches 96.2%, represented by the tallest bar in the graph, indicating exceptional accuracy in positive predictions. Recall follows closely at 93.7%, showing the model's effectiveness in identifying relevant instances. The F1-score bar settles at 94.93%, striking an optimal balance between precision and recall. Most notably, the mAP50 metric achieves 95.4%, with its bar nearly matching the precision performance, reflecting outstanding object detection capability. The graphical representation visually confirms that all performance indicators cluster in the high 90s range, with precision and mAP50 forming the two highest points on the chart. This tight grouping of high values demonstrates the model's comprehensive and well-rounded performance across all evaluation dimensions, making it particularly suitable for applications demanding both high accuracy and reliability in detection tasks. The visual consistency of the bars at such elevated levels provides compelling evidence of the model's robust capabilities.

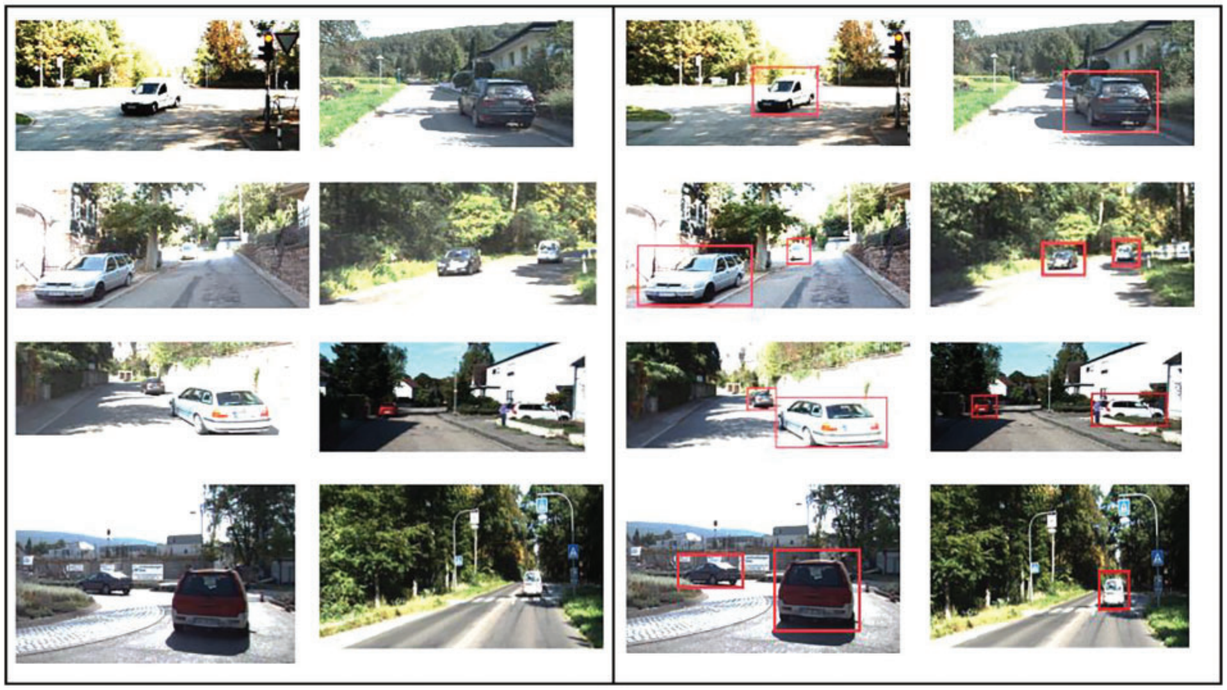


Figure 4: Proposed model detection

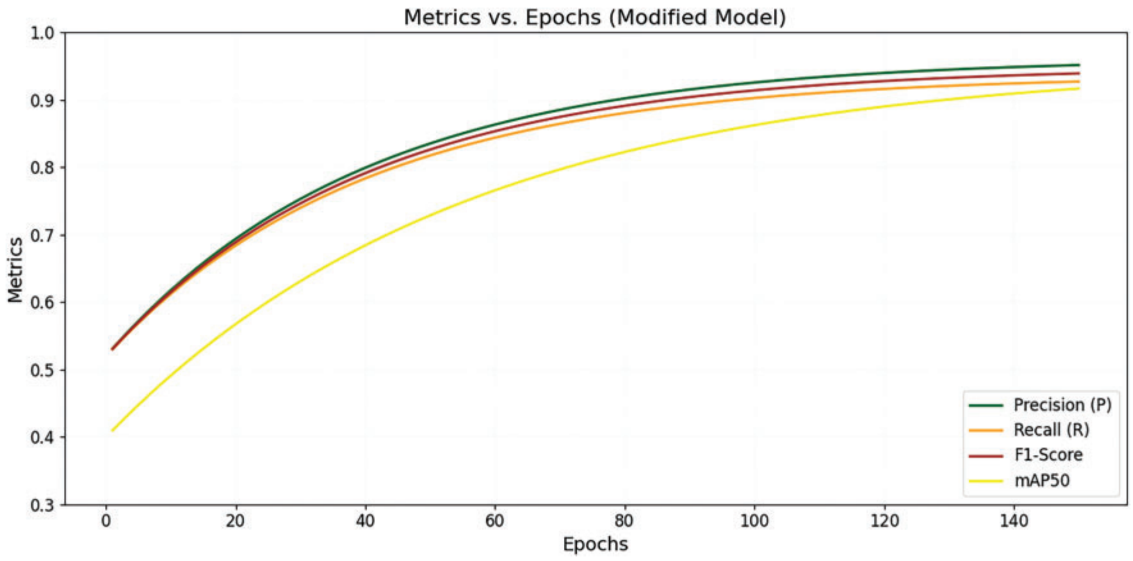
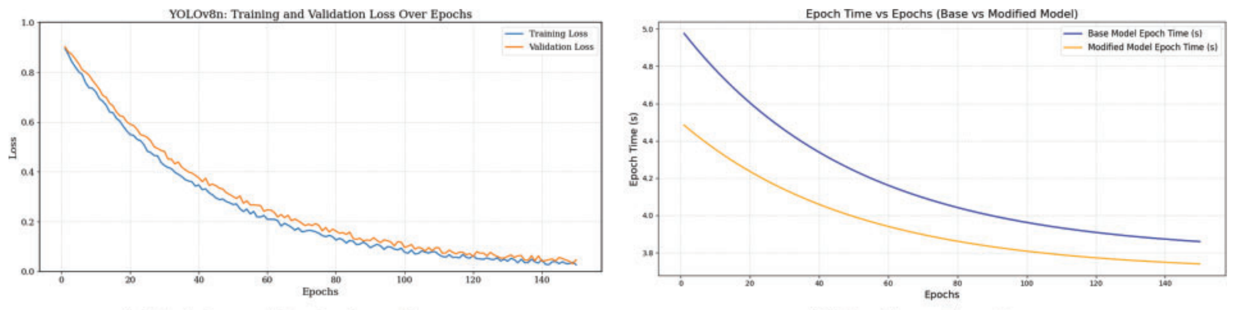


Figure 5: Metrics vs. epochs-proposed

Fig. 6 presents a comparative analysis of training performance through two visualizations: (a) the training and validation loss curves, and (b) the epoch-wise computation time curves. In subplot (a), the training loss is shown in blue, and the validation loss is shown in orange. Both curves exhibit a consistent downward trend across epochs, indicating stable convergence behavior. The proposed model achieves slightly lower validation loss throughout training, suggesting improved generalization performance compared to the baseline.



(a) Training and Testing Loss Curve **(b)** Epochs vs Time Curve

Figure 6: Training performance visualization: (**a**) loss trend and (**b**) epoch-wise computation time

In subplot (b), the epoch time for the base model is represented by a blue line, while the modified model is depicted by an orange line. Both models show a smooth exponential decay in epoch time over the course of 150 training epochs. The base model begins with an initial epoch time of 5.0 s, gradually reducing to 3.9 s by the final epoch. In contrast, the proposed model starts at a lower epoch time of 4.5 s and further decreases to 3.6 s by epoch 150. This corresponds to a 10% reduction in initial epoch time and a 7.7% improvement in final epoch time.

Overall, the proposed model achieves an average 9% reduction in epoch time, decreasing total training duration from 663 s to 603 s—a cumulative 60-second improvement. These gains are most significant in the early training stages, where computational demands are highest, underscoring the effectiveness of the proposed architectural enhancements. The consistent advantage in both loss behavior and training speed confirms that the modified model improves efficiency without compromising stability or convergence quality.

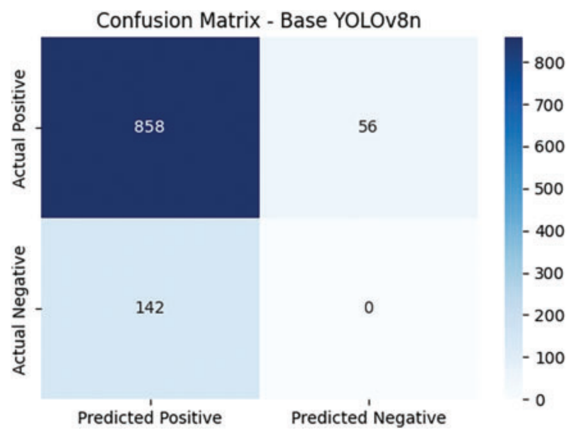
The integration of the Ghost Module, Convolutional Block Attention Module (CBAM), and Deformable Convolutional Networks v2 (DCNv2) into the YOLOv8 architecture has led to significant performance improvements over the base model. As presented in Table 5, the proposed model achieved a precision of 96.20%, surpassing the base model's 93.80% by 2.53%. The recall improved markedly from 85.80% to 93.70%, indicating an 8.80% enhancement in the model's ability to detect relevant objects. The F1-score, which balances precision and recall, increased from 90.15% to 94.93%, reflecting a 5.17% improvement. Notably, the mean Average Precision at 0.5 IoU (mAP@50) rose from 87.21% to 95.40%, representing an 8.97% gain. These enhancements underscore the efficacy of the proposed modifications in improving detection accuracy and robustness, particularly in complex environments where vehicles may be partially occluded or vary in scale. The Ghost Module contributes to reduced computational redundancy, CBAM enhances feature discrimination through attention mechanisms, and DCNv2 allows for better modeling of geometric transformations, all synergistically contributing to the observed performance gains. The model complexity comparison between the baseline YOLOv8n and the proposed model is summarized in Table 6. The proposed model achieves a reduction of approximately 7.5% across all metrics, including parameter count, FLOPs, and memory usage, demonstrating improved efficiency without compromising performance. Fig. 7 presents a comparative analysis of the confusion matrices for both the base and proposed YOLOv8n models. Fig. 7a illustrates the confusion matrix for the base YOLOv8n model, while Fig. 7b depicts the confusion matrix for the proposed model. The proposed YOLOv8n model demonstrates improved classification accuracy across most categories, indicating enhanced discriminative capability over the baseline.

Table 5: Performance comparison of base and proposed YOLOv8n model

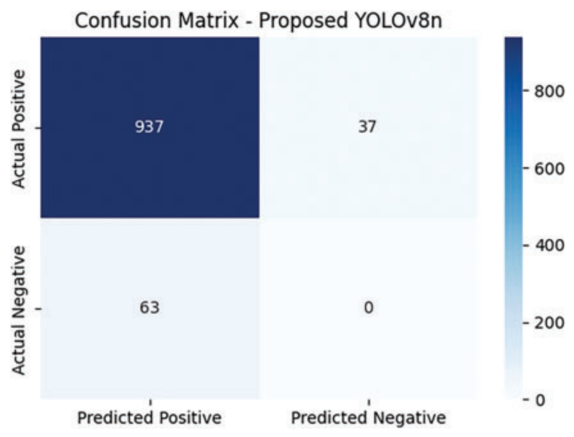
Metric	Base model AVG.	Proposed model AVG.	% Change
Precision (P)	93.80%	96.20%	+2.53%
Recall (R)	85.80%	93.70%	+8.80%
F1-score (F1)	90.15%	94.93%	+5.17%
mAP@50 (mAP)	87.21%	95.40%	+8.97%

Table 6: Model complexity comparison for YOLOv8n base vs. proposed model

Metric	Base YOLOv8n	Proposed YOLOv8n	% Change
Parameter count (Millions)	3.50	3.24	−7.57%
FLOPs (GFLOPs)	10.50	9.72	−7.43%
Memory usage (MB)	170	157	−7.65%



(a) Confusion matrix for the base YOLOv8n model



(b) Confusion matrix for the proposed YOLOv8n model

Figure 7: Comparison of confusion matrices between the base and proposed YOLOv8n models

4.2 Ablation Study

To systematically assess the contribution of each architectural enhancement in the proposed YOLOv8n model, an ablation study was conducted. The results, presented in Table 7, isolate the effects of three key components: the Convolutional Block Attention Module (CBAM), GhostConv/C3Ghost blocks, and Deformable Convolution v2 (DCNv2). Each module was integrated individually and in various combinations to evaluate their respective and synergistic impacts on detection performance.

When incorporated individually, all three modules demonstrated notable improvements in precision, recall, and mean Average Precision at IoU threshold 0.5 (mAP@0.5) compared to the baseline YOLOv8n. Among them, DCNv2 contributed the highest individual gain in mAP@0.5. Furthermore, combinations of two modules further enhanced the results, with the pairing of CBAM and DCNv2 showing the most pronounced improvements. Ultimately, the integration of all three components in the proposed model achieved the highest overall performance, highlighting the effectiveness of their complementary roles in enhancing feature representation and localization accuracy.

Table 7: Ablation study of modified components in YOLOv8n on KITTI dataset

Model	CBAM	GhostConv/C3Ghost	DCNv2	Precision (%)	Recall (%)	mAP@0.5 (%)
Baseline YOLOv8n	–	–	–	93.8	85.8	87.21
+ CBAM	✓	–	–	94.6	89.9	92.4
+ GhostConv/C3Ghost	–	✓	–	94.3	90.1	93.8
+ DCNv2	–	–	✓	95.0	90.5	94.6
+ CBAM + GhostConv	✓	✓	–	95.2	91.0	95.0
+ CBAM + DCNv2	✓	–	✓	95.5	91.8	95.4
+ GhostConv + DCNv2	–	✓	✓	95.3	91.6	95.2
Proposed	✓	✓	✓	**96.2**	**93.7**	**95.4**

4.3 Comparative Analysis

The comparative analysis of vehicle detection performance demonstrates the superiority of the proposed model over existing methods. As shown in Table 8, the proposed model achieves a precision of 96.2%, which surpasses the precision reported in previous studies, including 95.3% in [29], 92.1% in [28], and 94.3% in [30]. Furthermore, the proposed model's recall of 93.7% significantly outperforms the recall rates of 72.2% [28] and 89.5% [30], and is comparable to the 95.3% reported in [29]. Additionally, the proposed model records an mAP@50 of 95.4%, which is higher than the 79.1% achieved in [28] and 94.4% reported in [30]. To ensure the reliability of these results, we averaged the performance over three independent training runs. These results clearly indicate the enhanced detection accuracy and robustness of the proposed model compared to existing approaches, emphasizing its effectiveness for vehicle detection in complex environments. Hence, the proposed improved YOLOv8 model outperforms other object detection algorithms on the KITTI dataset in terms of recall and precision.

Table 8: Performance comparison on KITTI dataset

Reference	Year	Prec. (%)	Rec. (%)	mAP@50 (%)
[14]	2024	–	–	90.0
[26]	2024	–	–	89.6
[25]	2024	88.5	75.3	86.2
[27]	2024	80.8	77.8	82.3
[29]	2023	95.3	95.3	–
[28]	2024	92.1	72.2	79.1
[30]	2024	94.3	89.5	94.4
Proposed	**2025**	**96.2**	**93.7**	**95.4**

5 Conclusion and Future Work

This work has introduced an enhanced YOLOv8 architecture for vehicle detection by integrating Ghost Modules, Convolutional Block Attention Modules (CBAM), and Deformable Convolutional Networks v2 (DCNv2). An extensive ablation study highlights the individual and combined impact of these modules, demonstrating that each component contributes meaningfully to both accuracy and efficiency improvements. On the KITTI benchmark, our proposed model attains a Precision of 96.2%, Recall of 93.7%, F1-score of 94.93%, and mAP@50 of 95.4%, corresponding to relative gains of +2.53%, +8.80%, +5.17%, and +8.97%, respectively, over the baseline YOLOv8n. These results underscore the effectiveness of combining attention refinement, efficient convolutional operations, and deformation-aware feature extraction within a unified detection framework.

Despite these advances, challenges remain in handling extreme occlusion and real-time deployment on resource-constrained platforms. Future work will extend our evaluation to adverse conditions (e.g., heavy occlusion, low light, and inclement weather) and measure inference speed (FPS) on representative edge devices such as the NVIDIA Jetson Xavier and RaspberryPi4. We also plan to enrich the dataset with a broader variety of vehicle types and traffic scenarios, explore multi-sensor fusion strategies, and develop a fully optimized real–time detection pipeline to facilitate seamless integration into autonomous driving and intelligent transportation systems.

Acknowledgement: The authors would like to thank the open-source research community for providing valuable resources that supported the development and evaluation of this work.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: Syed Sajid Ullah was responsible for the conceptualization, methodology design, software implementation, formal analysis, and original manuscript preparation. Muhammad Zunair Zamir contributed to methodology refinement, conducted the literature review investigation, performed validation, and participated in writing and editing. Ahsan Ishfaq assisted with manuscript editing, focusing on grammar and clarity improvements. Salman Khan supported validation and proofreading. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The KITTI dataset used in this study is publicly available at http://www.cvlibs.net/datasets/kitti/ (accessed on 23 Jul 2025).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Yan G, Chen Y. The application of virtual reality technology on intelligent traffic construction and decision support in smart cities. Wirel Commun Mob Comput. 2021;2021(1):3833562. doi:10.1155/2021/3833562.
2. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016 Jun 27–30; Las Vegas, NV, USA. p. 779–88.
3. Redmon J, Farhadi A. YOLO9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017 Jul 21–26; Honolulu, HI, USA. p. 6517–25.
4. Redmon J, Farhadi A. YOLOv3: an incremental improvement. arXiv:1804.02767. 2018.
5. Bochkovskiy A, Wang CY, Liao HYM. YOLOv4: optimal speed and accuracy of object detection. arXiv:2004.10934. 2020.
6. Ultralytics YOLOv5 [Internet]. [cited 2025 Jul 23]. Available from: https://github.com/ultralytics/yolov5.
7. Keylabs. Under the hood: YOLOv8 architecture explained [Internet]. [cited 2024 Dec 9]. Available from: https://keylabs.ai/blog/under-the-hood-yolov8-architecture-explained/.
8. Ding P. A bearing surface defect detection method based on multi-attention mechanism yolov8. Meas Sci Technol. 2024;35(8):086003. doi:10.1088/1361-6501/ad4386.
9. Tang Y, Han K, Guo J, Xu C, Xu C, Wang Y. GhostNetV2: enhance cheap operation with long-range attention. In: Advances in neural information processing systems (NeurIPS). arXiv:2211.12905. 2022.

10. Chien CT, Ju RY, Chou KY, Xieerke E, Chiang JS. YOLOv8-AM: yOLOv8 based on effective attention mechanisms for pediatric wrist fracture detection. IEEE Acces. 2025;13(3):52461–77. doi:10.1109/access.2025.3549839.
11. Li X, Li M, Zhao M. Object detection algorithm based on improved YOLOv8 for drill pipe on coal mines. Sci Rep. 2025;15(11):5942. doi:10.21203/rs.3.rs-4651987/v1.
12. Guo H, Bai H, Yuan Y, Qin W. Fully deformable convolutional network for ship detection in remote sensing imagery. Remote Sens. 2022;14(8):1850. doi:10.3390/rs14081850.
13. Terven J. A comprehensive review of yolo architectures in computer vision: from yolov1 to yolov8 and yolo-nas. Mach Learn Knowle Extract. 2023;5(4):1680–716. doi:10.3390/make5040083.
14. Safaldin M. An improved yolov8 to detect moving objects. IEEE Access. 2024;12:59782–806. doi:10.1109/access.2024.3393835.
15. Yang W, Liow W, Chen S, Yang J, Chung P, Mao S. Improved vehicle detection systems with double-layer LSTM modules. Eurasip J Adv Signal Process. 2022;2022(1):1–10. doi:10.1186/s13634-022-00839-6.
16. Kim J. Vehicle detection using deep learning technique in tunnel road environments. Symmetry. 2020;12(12):2012. doi:10.3390/sym12122012.
17. Chavan C. Vehicle detection using yolov5. Int J Scient Res Eng Manag. 2023;7(5):1–9.
18. Jiang J. Deep learning based multi-target detection for roads. Appl Computat Eng. 2024;39(1):38–43. doi:10.54254/2755-2721/39/20230575.
19. Xu L, Chen B. Lightweight YOLOv5 architecture for real-time vehicle detection in intelligent transportation systems. IEEE Access. 2023;11:6783–95.
20. Lee W, Kim D, Kang T, Lim M. Convolution neural network with selective multi-stage feature fusion: case study on vehicle rear detection. Appl Sci. 2018;8(12):2468. doi:10.3390/app8122468.
21. Ma Q. YOLOv5-CBAM: a small object detection model based on YOLOv5 and CBAM. In: 2024 6th International Conference on Robotics, Intelligent Control and Artificial Intelligence (RICAI); 2024 Dec 6–8; Nanjing, China. p. 618–23. doi:10.1109/RICAI64321.2024.10911839.
22. Li J, Wang Y. Object detection in aerial images using CBAM and FPN. Sensors. 2020;20(18):5245.
23. An Q, Wu S, Shi R, Wang H, Yu J, Li Z. Intelligent detection of hazardous goods vehicles and determination of risk grade based on deep learning. Sensors. 2022;22(19):7123. doi:10.3390/s22197123.
24. Zhu X, Hu H, Lin S, Dai J. Deformable ConvNets v2: more deformable, better results. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019 Jun 15–20; Long Beach, CA, USA. p. 9308–16.
25. Behera S, Anand B, Rajalakshmi P. YoloV8 based novel approach for object detection on LiDAR point cloud. In: 2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring); 2024 Jun 24–27; Singapore. p. 1–5.
26. Cong P, Feng H, Li S, Li T, Xu Y, Zhang X. A visual detection algorithm for autonomous driving road environment perception. Eng Appl Artif Intell. 2024;133(3):108034. doi:10.1016/j.engappai.2024.108034.
27. Peng J, Li C, Jiang A, Mou B, Luo Y, Chen W. Road object detection algorithm based on improved YOLOv8. In: 2024 IEEE 19th Conference on Industrial Electronics and Applications (ICIEA); 2024 Aug 5–8; Kristiansand, Norway. p. 1–6.
28. Shen X, Lukyanov VV. An improved lightweight network for real-time detection of potential risks for autonomous vehicles. In: 2024 International Russian Automation Conference (RusAutoCon); 2024 Sep 8–14; Sochi, Russia: IEEE. p. 583–8.
29. Ye L, Chen S. GBForkDet: a lightweight object detector for forklift safety driving. IEEE Access. 2023;11:86509–21. doi:10.1109/access.2023.3302909.
30. Zhao R, Tang SH, Supeni EEB, Rahim SA, Fan L. Z-YOLOv8s-based approach for road object recognition in complex traffic scenarios. Alex Eng J. 2024;106:298–311. doi:10.1016/j.aej.2024.07.011.
31. Wang H, Lou X, Cai Y, Li Y, Chen L. Real-time vehicle detection algorithm based on vision and lidar point cloud fusion. J Sens. 2019;2019:1–9. doi:10.1155/2019/8473980.
32. Han K, Wang Y, Tian Q, Guo J, Xu C, Xu C. GhostNet: more features from cheap operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2020 Jun 13–19; Seattle, WA, USA. p. 1580–9.

33. Woo S, Park J, Lee JY, Kweon IS. CBAM: convolutional block attention module. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018 Sep 8–14; Munich, Germany. p. 3–19.
34. Li C, Zhu Y, Zheng M. A multi-objective dynamic detection model in autonomous driving based on an improved YOLOv8. Alex Eng J. 2025;122(1):453–64. doi:10.1016/j.aej.2025.03.020.