information

MDPI

Article

Enhanced Lightweight Object Detection Model in Complex Scenes: An Improved YOLOv8n Approach

Sohaya El Hamdouni *, Boutaina Hdioud and Sanaa El Fkihi

Advanced Digital Enterprise Modeling and Information Retrieval (ADMIR) Laboratory, Rabat IT Center, Information Retrieval and Data Analytics Team (IRDA), ENSIAS, Mohammed V University in Rabat, Rabat 11000, Morocco

* Correspondence: sohaya_elhamdouni@um5.ac.ma

Abstract

Object detection has a vital impact on the analysis and interpretation of visual scenes. It is widely utilized in various fields, including healthcare, autonomous driving, and vehicle surveillance. However, complex scenes containing small, occluded, and multiscale objects present significant difficulties for object detection. This paper introduces a lightweight object detection algorithm, utilizing YOLOv8n as the baseline model, to address these problems. Our method focuses on four steps. Firstly, we add a layer for small object detection to enhance the feature expression capability of small objects. Secondly, to handle complex forms and appearances, we employ the C2f-DCNv2 module. This module integrates advanced DCNv2 (Deformable Convolutional Networks v2) by substituting the final C2f module in the backbone. Thirdly, we designed the CBAM, a lightweight attention module. We integrate it into the neck section to address missed detections. Finally, we use Ghost Convolution (GhostConv) as a light convolutional layer. This alternates with ordinary convolution in the neck. It ensures good detection performance while decreasing the number of parameters. Experimental performance on the PASCAL VOC dataset demonstrates that our approach lowers the number of model parameters by approximately 9.37%. The mAP@0.5:0.95 increased by 0.9%, recall (R) increased by 0.8%, mAP@0.5 increased by 0.3%, and precision (P) increased by 0.1% compared to the baseline model. To better evaluate the model's generalization performance in real-world driving scenarios, we conducted additional experiments using the KITTI dataset. Compared to the baseline model, our approach yielded a 0.8% improvement in mAP@0.5 and 1.3% in mAP@0.5:0.95. This result indicates strong performance in more dynamic and challenging conditions.

Keywords: object detection; lightweight detectors; YOLOv8n; small object detection; occluded object detection; multiscale object detection

Academic Editor: Gholamreza Anbarjafari (Shahab)

Received: 12 August 2025
Revised: 18 September 2025
Accepted: 26 September 2025
Published: 8 October 2025

Citation: El Hamdouni, S.; Hdioud, B.; El Fkihi, S. Enhanced Lightweight Object Detection Model in Complex Scenes: An Improved YOLOv8n Approach. *Information* **2025**, *16*, 871. https://doi.org/10.3390/info16100871

1. Introduction

Currently, the digital world has led to a better understanding of human needs. In fact, humans have become increasingly demanding in terms of efficiency and effectiveness, driving the development of refined technologies that can meet these needs. With applications extending across various fields, such as autonomous driving and healthcare, the need for accurate and efficient systems is paramount. Among these technologies, object detection has emerged rapidly due to its capacity to recognize and locate objects in visual media. The capability of detecting objects has become increasingly useful in various fields, including healthcare [1,2], blind navigation systems, and traffic surveillance, among others. However,

it presents several challenges, such as the need for real-time accuracy and robustness in complex scenes [3].

Object detection has become a key area in computer vision, enabling the tracking and recognition of object categories in visual media. It significantly impacts the analysis and interpretation of visual scenes and is widely adopted in fields like healthcare, autonomous driving [4,5], and vehicle surveillance. Advancements in deep learning have driven improvements in the performance and adaptability of object detection systems, with these methods demonstrating strong generalization across various tasks. Many studies aim to enhance detection algorithms, continually pushing performance beyond traditional approaches.

Despite these advancements, deep learning methods continue to struggle with complex background scenarios, occlusions, and multiscale objects. In practice, complex models can hinder implementation, making it crucial to design deep learning algorithms that create fast and effective models. Numerous studies have concentrated on obtaining high accuracy or lightweight algorithms without achieving a balance between the two. That is why achieving lightweight solutions while maintaining the accuracy of the algorithms has emerged as a key research focus. Although the emergence of computer vision has led to certain challenges persisting, especially when dealing with complex scenes that contain small objects, occluded objects, and multiscale objects, in visual scenes, we often encounter small objects that are either naturally small or appear small in relation to their surroundings. Consequently, detecting small objects has become a significant and complex challenge due to their low resolution, limited number of pixels, and gradual diminishment of small object features during convolutional computation [6]. Moreover, the significant scale differences make it difficult to synchronize the transmission of features from objects of different sizes to deep networks. As a result, the detection model struggles to maintain robust scale invariance. Therefore, addressing multiscale variations in object detection remains a persistent challenge. Additionally, the presence of occluded objects further complicates detection due to the fragmentation of visual features, as these objects are partially hidden. This results in objects having complex shapes and appearances, which can obstruct the precise detection of occluded items. In fact, achieving improvements in both detection speed and accuracy is crucial.

In this study, we propose a low-parameter variant of YOLOv8n for object detection, evaluated on the KITTI and PASCAL VOC datasets. The model is efficient for real-time applications as its reduced parameter count leads to faster computation and lower memory consumption. In contrast to earlier detection techniques, the suggested method maintains a lightweight design with fewer parameters while achieving better performance in complicated situations. Among the contributions of this work are the following:

- A small object detection layer is designed to increase the number of detecting layers to four in order to optimize small objects' feature expression capability.
- The C2f-DCNv2 module leads in extracting the features of objects characterized by complicated forms and appearances, pushing the model to recognize occluded objects.
- At the neck section, the CBAM lightweight attention module is incorporated in order to help the model concentrate on informative features, consequently enhancing its capability to comprehend and interpret complex scenes.
- The GhostNet module relies on cheap operations to provide more feature maps, ensuring an accurate detection of objects, by decreasing the amount of parameters, thereby helping to improve the model's speed.

The remainder of this article is structured as follows: Section 2 reviews the state of the art; Section 3 describes our proposed method; Section 4 discusses the experimental results and analysis; and finally, Section 5 presents the conclusion and future work.

2. State of the Art

Object detection is a central task in computer vision, enabling the recognition and localization of objects within visual data. While deep learning-based methods have significantly improved detection performance, challenges remain in accurately identifying small, occluded, and multiscale objects, especially in real-time and resource-constrained scenarios.

Current methods are broadly categorized into one-stage and two-stage detectors. One-stage approaches, such as YOLO [7] and SSD [8], perform object classification and bounding box regression in a single step, offering high speed suitable for real-time applications. Two-stage methods, including R-CNN [9] and Fast R-CNN [10], first generate region proposals before classification, typically achieving higher accuracy but at the expense of increased computational cost.

To address the limitations of detecting small or occluded objects, especially in lightweight settings, several improved YOLO-based models have been proposed. SO-YOLOv8 [11] enhances YOLOv8 with SE blocks, multiscale training, and data augmentation to improve performance on small object detection. TA-YOLO [12] introduces an MCSTA module to boost feature representation in remote sensing imagery while maintaining a lightweight structure. LAYN [13] combines GhostNet and attention-based fusion to reduce model complexity and improve detection on embedded devices. YOLO-MS [14] improves multiscale representation through varied convolutional kernels, serving as an efficient and modular enhancement to YOLO models. VFN [15] incorporates a multiscale Swin Transformer and vision enhancement modules to recover lost features in occluded object detection.

While these methods demonstrate significant progress, they typically address only one specific challenge, such as small object detection or occlusion, and they still face trade-offs between detection accuracy and computational efficiency in real-time scenarios. In contrast, our proposed method offers a more holistic and balanced design by simultaneously targeting multiple challenges within a lightweight architecture. Compared to SO-YOLOv8 [11] and TA-YOLO [12], which primarily focus on enhancing small objects using attention mechanisms or advanced data augmentation, our approach introduces a dedicated small object detection layer and the C2f-DCNv2 module, which is specifically tailored to handle both small and occluded objects. Unlike LAYN [13], which relies solely on GhostNet for reducing complexity, we further enhance feature refinement by integrating a CBAM attention module in the neck. By combining GhostNet with these targeted modules, our method achieves a lower parameter count while preserving high detection accuracy. Evaluated on the KITTI and PASCAL VOC datasets, our model consistently outperforms these lightweight counterparts in both accuracy and inference speed, demonstrating its robustness and practical suitability for deployment in resource-constrained and real-time environments.

3. Proposed Method

In recent research, there has been a strong interest in the YOLOv8 algorithm due to its ability to detect objects quickly and efficiently, with high accuracy and good flexibility. The literature on YOLOv8 shows a variety of versions: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x, etc.

The findings suggest that, as a foundational model for high detection performance and speed, the YOLOv8n model may be helpful. In contrast to our method, which may enhance the performance of the YOLOv8n model, we first add a small object detection layer, increasing the total number of detection layers to four, and improve the model's ability to detect small objects. Secondly, to effectively extract the features of objects characterized by complicated forms and appearances, in the backbone, we replace the last C2f layer with the C2f-DCNv2 module. Thirdly, we designed the CBAM lightweight attention module to

address missed detections by integrating it into the neck section. Previous studies indicate that both the channel and spatial attention modules in the CBAM lightweight attention module improve the ability to understand and interpret complex scenes. Finally, to ensure good detection performance and reduce parameters in our enhanced method, GhostConv is used as a lightweight convolutional layer to alternate with ordinary convolution in the neck section, thereby maintaining good detection performance. In Figure 1, the complete structure of the changed YOLOv8n model is shown.

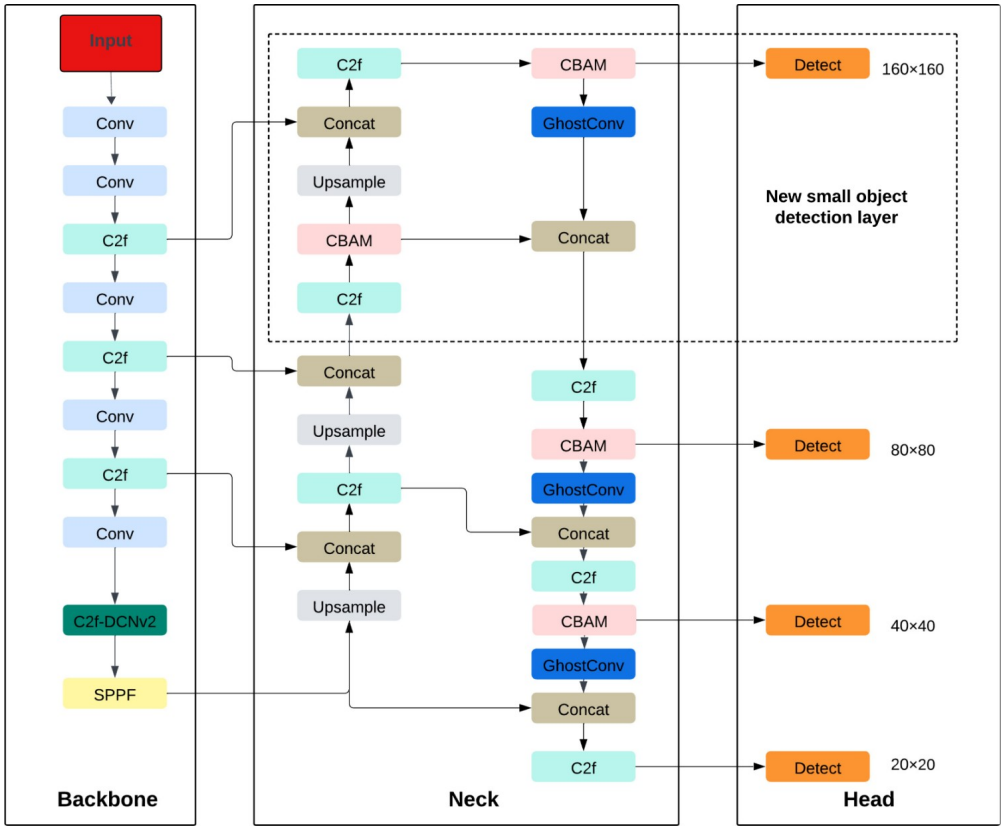

Figure 1. Enhanced YOLOv8n architecture with four-layer detection head, C2f-DCNv2, CBAM, and GhostConv integration.

The enhanced model is explained in detail in the next subsections.

3.1. Adding Small Object Detection Layer

When scenes are complex and contain intricate objects, object detection becomes essential. Object detection models, such as YOLOv8n, are efficient at improving accuracy and balancing speed, making them suitable for real-time scenarios. Conversely, the YOLOv8n model, which may struggle with complex scenes containing different object scales due to its high down-sampling factor, detects small objects poorly [16].

In fact, with the YOLOv8n model, 640 × 640 represents the dimensions (width and height) of the input image; however, the smallest size at which an object can be efficiently detected is 80 × 80. Summing up the YOLOv8n treatment, it can be concluded that each grid cell has a dimension of 8 × 8. A key limitation of that is when the objects are smaller than 8 pixels; therefore, YOLOv8n faces a significant challenge [17]. To overcome this problem, we utilize a small object recognition layer, sized 160 × 160.

In this adopted technique, the number of detection heads is raised to four. Together, those four layers, in collaboration with the other three layers, allow for the generation of more detailed feature information specific to various scales [18], which is also useful in mitigating the negative impact of variations in object dimensions. To improve the

model's detection performance, consider incorporating a small object detection layer. In this proposed technique, the upward upsampling feature layer in the neck section was combined with the 80×80 scale feature layer that was extracted from the fifth layer of the backbone network. The layer remains stacked with the superficial positional feature layer from the third layer in the backbone network. This supplementation enhances the formulation capability of the 160×160 scale fusion feature layer. In practical terms, this combination is subjected to processing by the C2f module and upsampling operations.

Upon integrating a small object detection layer, the C2f refines the identification process for small objects by passing the refined features to an extra decoupled head in the head section. We can obtain an enhanced detection method by enabling the feature representations of small objects, which are then transferred to the feature layers of the other three scales along the down-sampling path through the head section. This enhances the network's feature fusion capabilities and improves its accuracy in identifying small objects. The dashed part in Figure 1 shows the fourth layer, which has been added for small object detection.

3.2. C2f-DCNv2: Improved C2f Module with Deformable Convolution Networks v2

As shown below, object detection requires a robust backbone network to generate rich features [19]. Furthermore, deformable convolution is not fixed and can dynamically adjust, contrary to standard convolution [20]. In addition to the standard offset, it provides learnable offsets. Bilinear interpolation can be employed to determine the locations of pixels and calculate the associated feature values since offsets may be fractional.

Equation (1) presents the formula of deformable convolution:

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n + \Delta p_n), \tag{1}$$

where

- $\mathcal{R}$: the regular receptive field

$$\mathcal{R} = \{(-1,-1,-1),(-1,0,0),\ldots,(0,1,1),(1,1,1)\}; \tag{2}$$

- $w(p_n)$: the convolutional kernel's weight values at position p_n;
- $x(p_0 + p_n + \Delta p_n)$: the feature value of the feature map at location $p_0 + p_n + \Delta p_n$;
- Δp_n: the offset that is applied to the position $p_0 + p_n$.

For several years, considerable effort has been devoted to studying the use of deformable convolution networks, which has allowed researchers to cover objects at different scales. In fact, a deformable convolution network enhances the model's feature representation capability, thereby increasing the detection performance [20]. A key limitation of a deformable convolution network is that it may cover irrelevant regions, thereby decreasing the total performance. Ref. [21] introduced DCNv2, an improved version of deformable convolution, as a solution to this challenge.

By including $\Delta m_k \in [0,1]$ and determining the weights of sampling points, DCNv2 implements a modulation mechanism. Δm_k represents a low benchmark for uninteresting areas [21]. Although more learning parameters are required, the model's improvement makes this strategy extremely valuable.

The equation that describes DCNv2 is as follows:

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n + \Delta p_n) \cdot \Delta m_k. \tag{3}$$

The literature on the YOLOv8n model shows that the backbone section is leveraged to more accurately recognize images. In practice, YOLOv8n interprets images and elevates performance by using multiple convolutional layers. These layers combine features from different levels to ultimately extract features at various levels.

Moreover, fixed standard convolution kernels exhibit limited flexibility, resulting in restricted receptive fields that capture only local object information [22]. Because of this, the network misses or falsely detects objects when encountering complex scenes, such as occluded ones, small objects [23], or multiple objects.

One of the main advantages of incorporating DCNv2 into the backbone section of YOLOv8n is its ability to optimize the receptive field, allowing the model to capture objects with varying shapes, scales, and partial occlusions more effectively. Figure 2 illustrates the architecture of the C2f-DCNv2 module, which is built upon the Bottleneck-DCNv2 structure. The lower part of the figure details two variants of the Bottleneck-DCNv2 block: with Shortcut = True, where the input is added to the output of two sequential DCNv2 layers through a residual connection, facilitating gradient flow and promoting feature reuse; and with Shortcut = False, where this residual path is omitted and only two DCNv2 layers are stacked in sequence, focusing on nonlinear transformations while reducing structural complexity. The upper part of the figure shows the overall C2f-DCNv2 module: after an initial convolution, the feature maps are split and processed through n Bottleneck-DCNv2 blocks. The outputs are then concatenated and passed through a final convolutional layer, enabling multiscale feature extraction and enhancing the model's capability to capture fine-grained details in complex shapes.

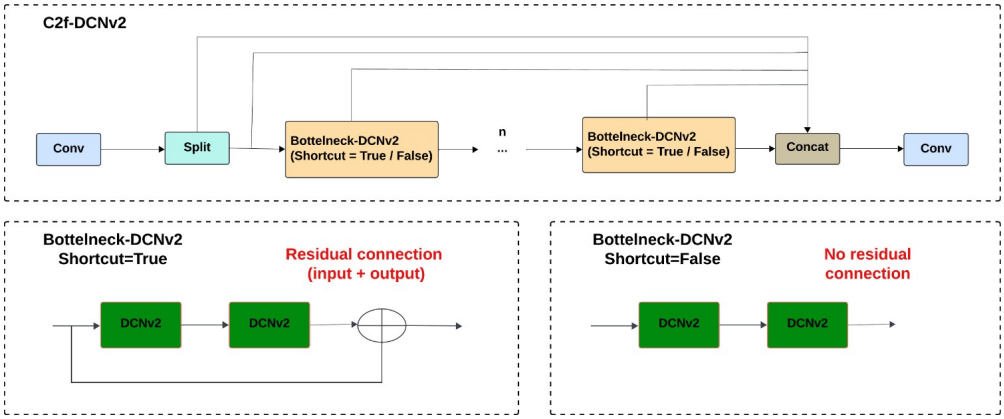

Figure 2. The structure of the C2f-DCNv2 module.

The findings indicate that the C2f-DCNv2 module not only optimizes the receptive field, but also contributes to improved detection performance. Nevertheless, it is important to note that these benefits come at the cost of reduced inference speed and increased parameter tuning complexity. To balance these trade-offs, further experiments are required to evaluate detection accuracy alongside runtime efficiency. Overall, our results demonstrate that replacing the final C2f module in the backbone with the C2f-DCNv2 module yields optimal performance gains.

3.3. The CBAM Lightweight Attention Module

Incorporating attention mechanisms into object detection tasks improves the network's focus on objects, enhancing detection performance in images by assigning more weight to relevant information and ignoring non-essential information. This section presents the addition of the Convolutional Block Attention Module (CBAM) to the YOLOv8n neck section, as illustrated in Figure 1. In fact, the CBAM aims to address small objects in complex scene perception. The delicate characteristics of small objects are the main disadvantage.

In this context, complex scene perception involves difficulties such as interactions between multiple objects, scale variations, and occlusions. From this, we conclude that the CBAM lightweight attention module enables the model to focus on key features, thereby enhancing its ability to better understand and interpret complex scenes.

In Figure 3, the CBAM network has a dual structure, including the channel attention module (CAM) and the spatial attention module (SAM) [24]. In fact, the CBAM lightweight attention module facilitates the network by locating precise and specific information about the area. Max-pooling and average-pooling operations are applied to the input feature map [24], designated as F, by the first CAM module to start the CBAM module. A shared Multilayer Perceptron (MLP) neural network then processes the outputs of the two pooling operations. Moreover, a feature map M_c produced by the activation of the sigmoid function is the MLP's outcome.

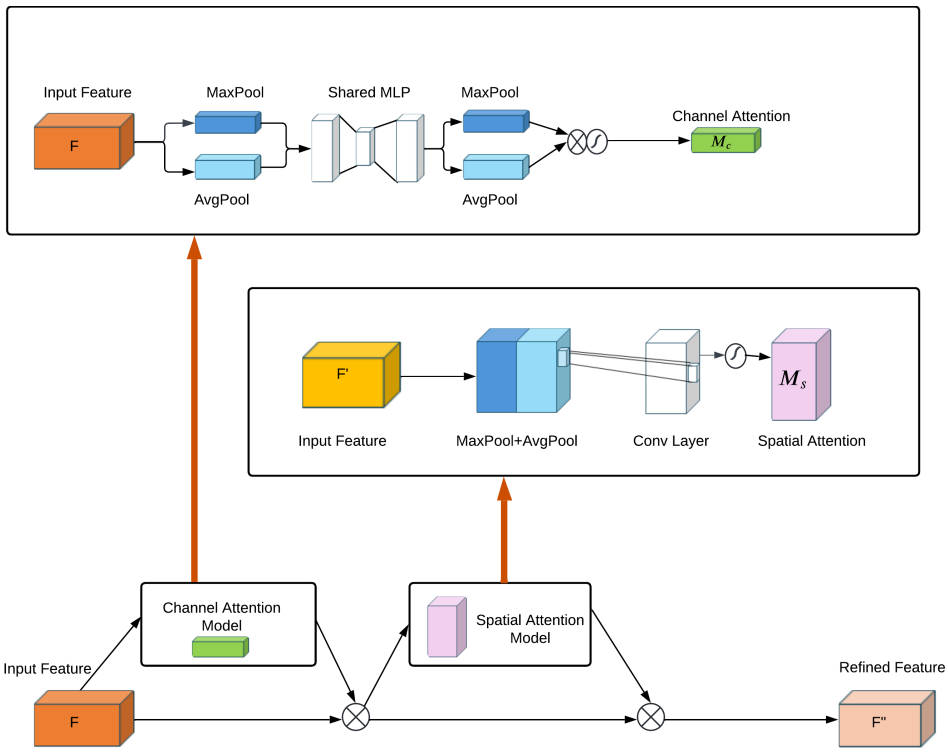

Figure 3. CBAM network structure diagram [24].

Equation (4) demonstrates the expression of the channel attention:

$$\begin{aligned} M_c(F) &= \sigma(\mathrm{MLP}(\mathrm{AvgPool}(F)) + \mathrm{MLP}(\mathrm{MaxPool}(F))) \\ &= \sigma\Big(W_1\Big(W_0(F^c_{\mathrm{avg}})\Big) + W_1(W_0(F^c_{\mathrm{max}}))\Big), \end{aligned} \tag{4}$$

where

- σ: the sigmoid activation function;
- W_0 and W_1: the weights of the MLP;
- $\mathrm{F}^c_{\mathrm{avg}}$: the average-pooled of feature F;
- $\mathrm{F}^c_{\mathrm{max}}$: the max-pooled of feature F.

To obtain the transformed input feature F' needed for the SAM, the input feature map F is then multiplied by M_c.

$$F' = M_c(F) \otimes F. \tag{5}$$

From the CAM output feature map F', average- and max-pooling are applied along the channel axis, and the results are concatenated to create a robust feature descriptor, which is

followed by a 7×7 convolution and then by a sigmoid activation to produce the spatial attention map $M_s(F')$ (the spatial characteristics M_s focus on localizing information [25]). Equation (6) expresses the spatial attention:

$$\begin{aligned} M_s(F') &= \sigma\Big(f^{7\times7}\big([\text{AvrPool}(F');\text{MaxPool}(F')]\big)\Big) \\ &= \sigma\Big(f^{7\times7}\Big[F^s_{\text{avg}};F^s_{\text{max}}\Big]\Big), \end{aligned} \tag{6}$$

where

- σ: the sigmoid activation function;
- $f^{7\times7}$: refers to a convolution with a filter size of 7×7;
- F^s_{avg}: the average-pooled of feature F';
- F^s_{max} : the max-pooled of feature F'.

The adjusted input feature map F' is multiplied by the spatial feature map M_s to produce the final feature map F'', as shown in the following equation:

$$F'' = M_s(F') \otimes F'. \tag{7}$$

3.4. Integrating the GhostNet Module

After applying several techniques to enhance our model, decreasing the number of parameters can enhance the model's efficiency, making it particularly suitable for deployment on embedded or resource-limited systems. Most state-of-the-art methods rely on deep convolutional neural networks (CNNs), which typically combine multiple types of convolutional operations to extract rich and diverse feature representations, resulting in massive computational costs [26]. Figure 4a illustrates how output feature maps from convolutional layers frequently have a great deal of redundancy and may even imitate others. According to [27], to simplify processing, in 2020, Huawei Noah's Ark Laboratory developed GhostNet, a lightweight network. Moreover, the GhostNet network contains a convolutional module called GhostConv that can replace conventional convolutions. The GhostConv module splits the traditional convolution into two sections. The first section is about ordinary convolutions; however, there will be strict limits on the overall number of convolutions. Instead of altering the size of the resulting feature map, a sequence of general linear operations is then performed on the intrinsic feature maps created in the first section to produce additional feature maps. To obtain the final results, Figure 4b illustrates how these two types of feature maps are combined. According to [28], GhostConv significantly reduces the number of parameters and computational load while maintaining model performance, enabling efficient and lightweight deployment of neural networks, thereby balancing speed.

Equations (8) and (9) reveal that the ratio of FLOPs to parameters is influenced by the transformation count s.

$$\begin{aligned} r_s &= \frac{h \times w \times c \times H \times W \times n}{\frac{n}{s} \times H \times W \times k^2 \times c \times (s-1) \times \frac{n}{s} \times H \times W \times d^2} \\ &= \frac{c \times k^2}{\frac{1}{s} \times c \times k^2 + \frac{(s-1)}{s} \times d^2} \\ &\approx s, \end{aligned} \tag{8}$$

$$r_c = \frac{n \times c \times k^2}{\frac{n}{s} \times c \times k^2 + (s-1) \times \frac{n}{s} \times d^2} \approx \frac{s \times c}{s + c - 1} \approx s. \tag{9}$$

Let

- r_s: the ratio of floating-point operations (FLOPs);
- r_c: the parameter ratio between standard convolutional layers (Conv) and Ghost Convolution layers (GhostConv);
- h and w: the height and width of input features;
- c: the channel number of input features,
- H and W: the height and width of output features;
- n: the number of convolutional kernels;
- $k \times k$: the convolution kernel's size;
- d: the linear transformation kernel size;
- s: the transformation count.

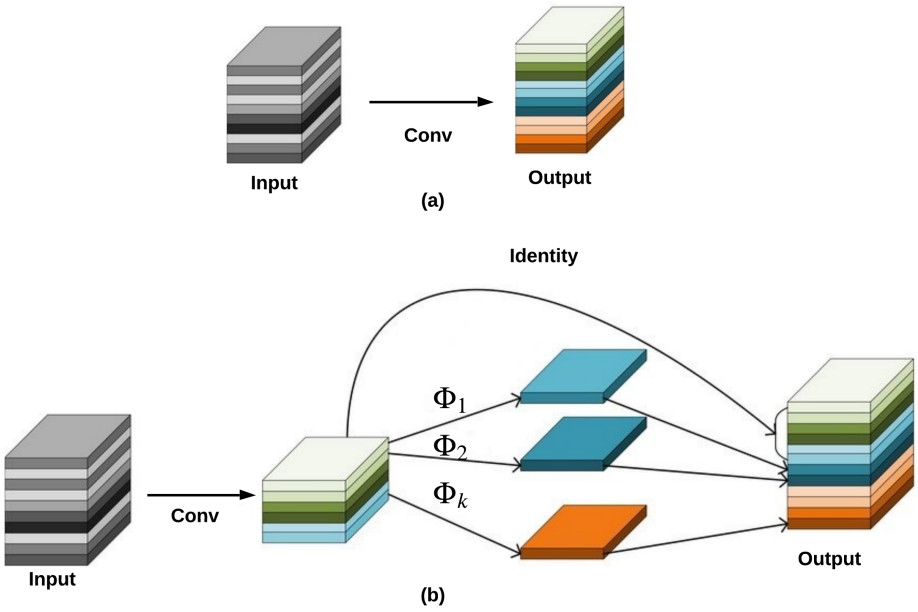

Figure 4. (**a**) Architecture of the standard convolution; (**b**) architecture of GhostConv [27].

Essentially, the more feature maps are generated, the more the model is accelerated. Furthermore, integrating GhostConv into the model leads to an efficient technique for decreasing the number of parameters, which will ultimately improve the model's efficiency and operating speed. By replacing the convolution with GhostConv in the neck section of YOLOv8n, we deduce that substituting the original network modules with GhostConv leads to a lighter and faster network, which compresses the network model size, thereby decreasing the number of parameters.

To ensure transparency and reproducibility, the implementation of our proposed model, including the C2f-DCNv2 module and the integration of GhostConv, has been made publicly available at https://github.com/Sohaya-ELHAMDOUNI/YOLOv8n-Enhancement-for-Complex-Scene-Detection, accessed on 25 September 2025.

4. Experimental Results and Analysis

To demonstrate the effectiveness of our model, we employed two benchmark datasets commonly used in object detection research: PASCAL VOC and KITTI. Experiments were performed to evaluate the proposed approach and to benchmark it against YOLOv8 and other leading methods. The results reveal that our model delivers superior performance, particularly in handling object detection within complex environments. The following section outlines the main aspects of the model's implementation, describes the datasets applied, specifies the evaluation metrics, and presents the experimental outcomes.

4.1. Datasets

The Pascal Visual Object Classes (VOC) dataset, as proposed in [29], was first created at the University of Oxford Computer Vision group in the United Kingdom. PASCAL VOC has gained popularity as a standard for evaluating object detection performance, pushing researchers each year to improve it and overcome their limits. A wide range of variations is present in the dataset, including differences in object size, orientation, pose, lighting, location, and occlusion, representing varied visual content. The dataset contains a rich set of annotated images with bounding boxes, including 20 object categories related to human life, involving humans; animals (cow, cat, bird, sheep, horse, and dog); vehicles (aeroplane, bicycle, boat, bus, train, motorbike, and car); and indoor items (chair, bottle, sofa, potted plant, table, and television).

The images are of varying sizes and aspect ratios, and the annotations are in the PASCAL VOC format (XML), which was converted to the YOLO format to ensure compatibility with our training pipeline. In practical terms, a single training set comprising 16,551 samples is created by combining the training and validation data from the two years, 2007 and 2012. Additionally, the 2007 test data, consisting of 4952 samples, is used as the validation or test set. Recently, considerable attention has been paid to the PASCAL VOC dataset, leveraging its capabilities to advance similar works. On the other hand, the PASCAL VOC dataset includes small, intermediate, and large objects. The distribution of instance sizes in the PASCAL VOC dataset is seen in Table 1, indicating that 14% of instances have a size smaller than 50 pixels, 61% of instances have a size between 50 and 300 pixels, and 25% of instances have a size greater than 300 pixels. The visualization of the dataset is illustrated in Figure 5.

Table 1. Distribution of the instance sizes in the PASCAL VOC dataset.

Size (Pixels)	**<50**	**50–300**	**>300**
Percentage	14%	61%	25%

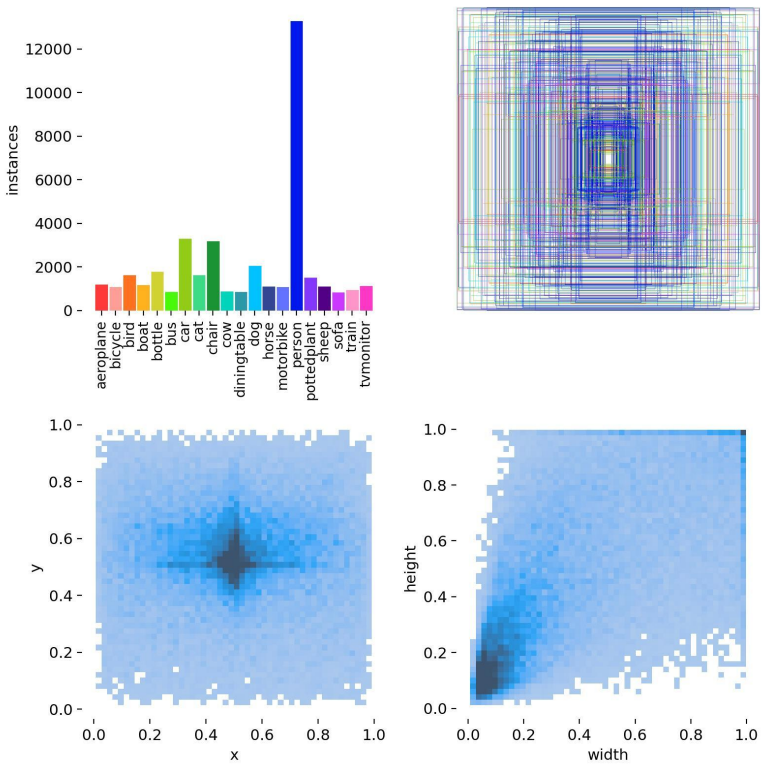

Figure 5. The dataset's illustration.

The KITTI [30] dataset, a benchmark extensively employed in autonomous driving studies for object detection, was recorded using a moving platform while driving in and around Karlsruhe, Germany. It includes camera images, laser scans, high-precision GPS measurements, and IMU accelerations collected through a combined GPS/IMU system [31]. The dataset includes 7481 images depicting real-world traffic scenarios, each annotated with one of nine key object types. These encompass various kinds of vehicles (e.g., cars, vans, trucks, and trams), human-related classes (pedestrians, persons, and cyclists), a miscellaneous category, and a class reserved for non-relevant or ignored objects. For the purpose of this study, the dataset was reorganized and divided into 80% for training (5984 images) and 20% for validation (1497 images). To ensure compatibility with our model training pipeline, the original KITTI annotations were converted into a standard format. This conversion preserved important information, such as object classes and 2D box coordinates, enabling fast data processing and model input. Notably, most objects in KITTI are small, especially in cases involving fast-moving or complex scenes, making it particularly well suited for testing detection model performance on small object tasks in autonomous driving scenarios. Additionally, the dataset presents challenges due to low-contrast targets and occlusions in dense environments, which can significantly impact detection accuracy. The KITTI dataset is made available for academic use only and requires approval for access, as outlined on the official KITTI website https://www.cvlibs.net/datasets/kitti/, accessed on 25 September 2025.

To enhance dataset diversity and model robustness in complex scenes, this study employed general data augmentation techniques, implemented in YOLOv8, for two datasets: PASCAL VOC and KITTI. These techniques included random erasing, horizontal flipping, translation, HSV adjustments (hue, saturation, and value), cropping, and scaling. Furthermore, we employed Mosaic augmentation, which integrates four separate training images to create a single composite, significantly improving context understanding and helping the model recognize small or occluded objects more effectively. The application of these techniques resulted in a significant enhancement of the model's generalization. Data augmentation also reduces overfitting by introducing variability to the training data, making it less likely that the model will memorize specific image features. Through these preprocessing and augmentation processes, we created optimized datasets that served as a solid basis for training and evaluating performance.

These two datasets enable us to verify the detection performance of our model in both general computer vision environments and autonomous driving scenarios. For the PASCAL VOC dataset, the network has to contend with a greater number of object classes and more complex backgrounds. In the case of the KITTI dataset, we consider small objects, such as pedestrians and vehicles in autonomous driving, as indicators of model performance in complex driving scenarios, providing a tougher test of its ability to generalize in such scenarios.

4.2. Evaluation Metrics

In our experimentation, we included several evaluation metrics, including R, P, mAP, model parameters (*Params*), and Giga Floating Point Operations Per Second (*GFLOPs*). In fact, mAP represents the average precision AP computed for each class, based on different IoU thresholds that define valid detections [19]. Specifically, mAP@0.5 is calculated at an IoU threshold of 0.5, while mAP@0.5:0.95 averages the results across thresholds from 0.5 to 0.95 in steps of 0.05. Equations (10)–(13) determine P, R, AP, and mAP. The higher the mAP is, the more accurate the model's object detection capabilities become.

$$P = \frac{TP}{TP + FP}, \tag{10}$$

$$R = \frac{TP}{TP + FN}, \tag{11}$$

$$AP = \int_0^1 P(R)dR, \tag{12}$$

$$mAP = \frac{\sum_{i=1}^{N} AP_i}{N}, \tag{13}$$

where

- TP: True Positive, demonstrating the results when the predicted and the sample are positive;
- FP: False Positive, proving that although the prediction is negative, the sample is positive;
- FN: False Negative, demonstrating that although the prediction is negative, the actual sample is positive;
- AP: the average precision;
- mAP: the average AP values calculated for each class in given dataset;
- N: the number of categories.

4.3. Experimental Environment and Hyperparameter Settings

Table 2 presents comprehensive details regarding the experimental configuration.

Table 2. Experimental configuration.

Component	Details
GPU	Tesla V100 SXM2
VRAM	16 GB
Python	3.10.12
Framework	Ultralytics YOLOv8.1.6
DL Architecture	PyTorch 2.2.1 + cu121

Table 3 lists the training parameters. The input sample sizes in this experiment were normalized to 640 × 640. The training used the SGD optimizer, with momentum set to 0.937, weight decay to 0.0005, and an initial learning rate of 0.01 to avoid overfitting. The batch size was fixed at 16 and training runs for 200 epochs, allowing for early stopping with a patience of 50 epochs. All other hyperparameters remained at the default YOLOv8n values.

Table 3. Model training settings.

Hyperparameters	Values
Epochs	200
Patience	50
Batch size	16
Optimizer	SGD
Momentum	0.937
Weight decay	0.0005
Initial learning rate	0.01
Image size	640 × 640

4.4. Experimental Results on PASCAL VOC Dataset

As shown in Figure 6, the SGD optimizer improves the model by adjusting the weights and other parameters, as evidenced by the results of the loss functions, which minimize as the training process progresses. We find that the loss functions rapidly declined before reaching 100 epochs, but the precision, recall, mAP@0.5, and mAP0.5:0.95 all showed

improvements. The level at which the loss function values progressively decreased was after 120 epochs. A deceleration was also evident in the increases in recall, accuracy, mAP@0.5, and mAP@0.5:0.95. Our technique yields strong results, indicating convergence. Upon reaching 200 epochs, the training loss curve showed little decline, and the other metric values stabilized. After a brief overview of the findings, it can be stated that the model is effective because the appropriate network weights were determined.

We aimed to provide a comprehensive representation of our experimental results. In Table 4, the precision, recall, mAP@0.5, mAP@0.5:0.95, Params, and FLOPs were compared to those of the YOLOv8n model. Regarding precision, our model achieved 82.5%, which is a 0.1% increase compared to the original YOLOv8n model. The recall of our model improved to 76.7%, showing an increase of 0.8%. Furthermore, the mAP@0.5 metric for the constructed model reached 83.9%, representing a 0.3% improvement. The mAP@0.5:0.95 of our proposed model was 64.4%, which is 0.9% higher than the original model. Additionally, based on the experimental findings, it can be stated that the research was highly successful, resulting in a 9.37% reduction in the number of parameters. The findings show that, compared to the original YOLOv8n model, our approach enhances the output, reduces the model size, and improves model accuracy. To investigate the particular performance of our method, we conducted a series of detailed tests.

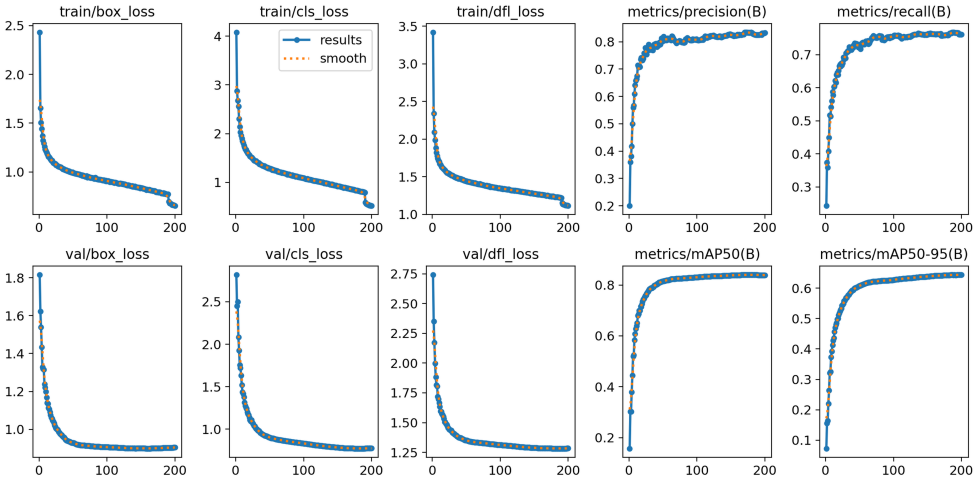

Figure 6. The training results of our proposed model.

Table 4. Comparative analysis of the detection accuracy.

Methods	P (%)	R (%)	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Params (M)	FLOPs (G)
YOLOv8n	82.4	75.9	83.6	63.5	3.2	8.9
Ours	82.5	76.7	83.9	64.4	2.9	11.9

Figure 7 shows the precision–recall curve for the 20 different categories in the PASCAL VOC dataset. When 0.5 is chosen as the *IOU* threshold, the results are shown by these curves, which is followed by calculating mAP through the total area under each curve. The horse class achieves the highest performance with an mAP@0.5 of 93.2, while the potted plant class has the lowest with an mAP@0.5 of 59.1. The all classes' curve reached a mAP@0.5 of 83.9. In fact, the precision–recall (PR) curve is a detailed curve that plots precision, which evaluates the accuracy of positive predictions, versus recall, which indicates the model's ability to correctly identify actual positives. A precision–recall curve provides a view of the model's performance, especially when the cost of false positives and false negatives varies.

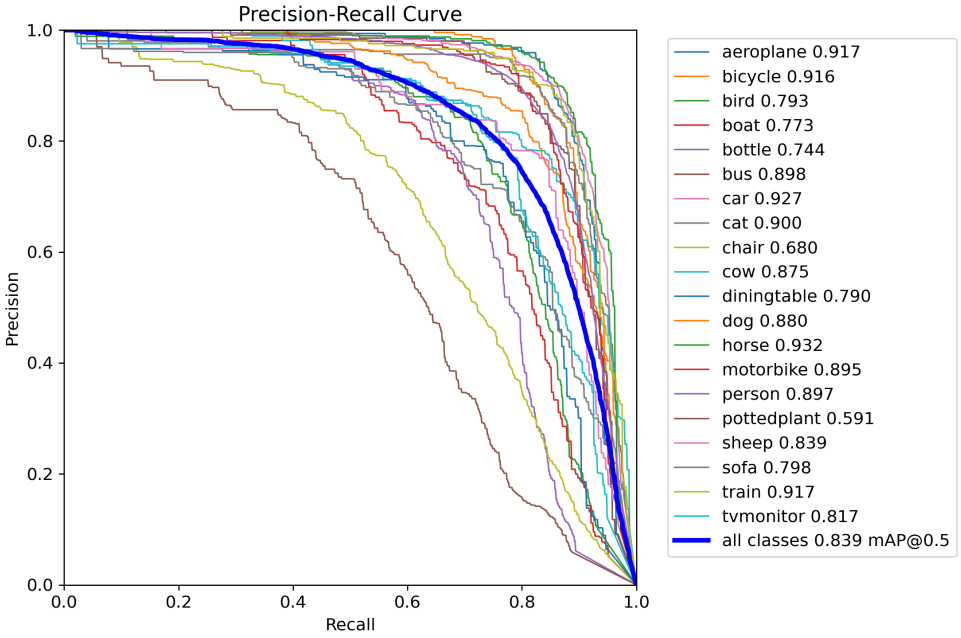

Figure 7. Precision–recall curves for the 20 classes on the PASCAL VOC dataset.

In Figure 8, the normalized confusion matrix is displayed. Each row in this matrix reflects the actual attributed category of the data, whereas each column indicates the anticipated category. The confusion matrix's diagonal values indicate the proportion of accurate predictions for each category, and they are mostly used to compare the expected and actual values.

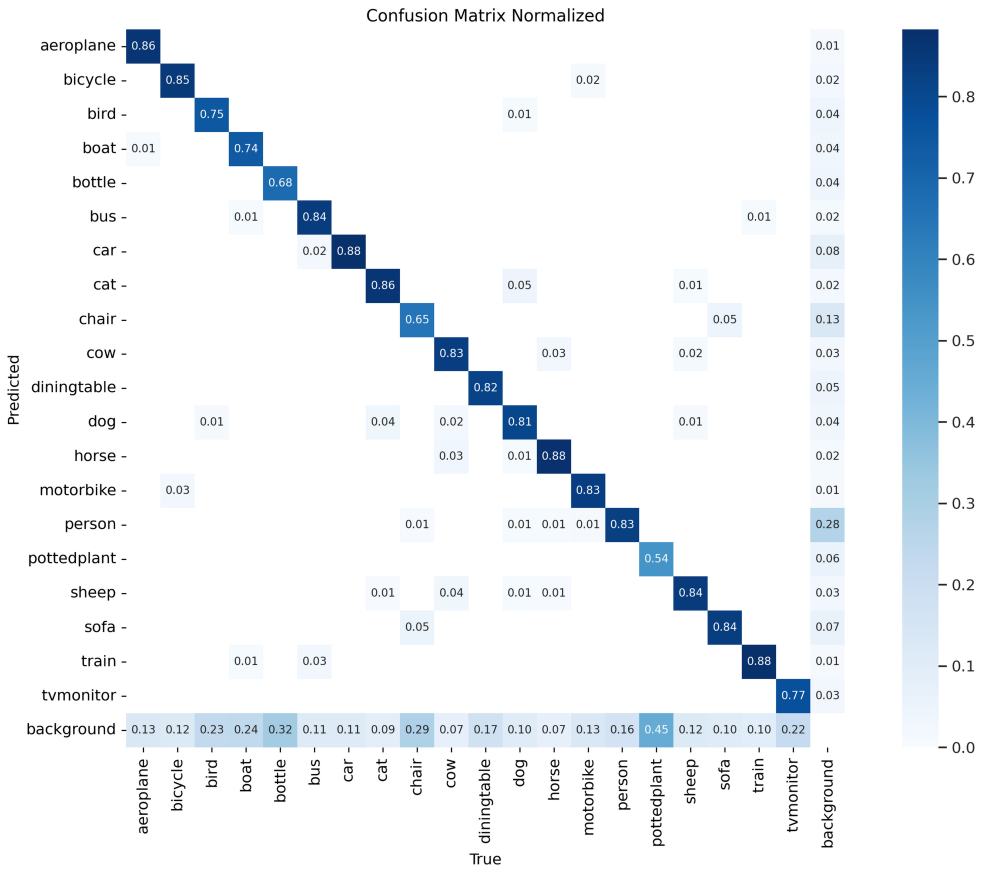

Figure 8. Normalized confusion matrix diagram.

For further experiments, we selected three categories of classes that contain both small objects and occluded objects in the PASCAL VOC datasets. For example, some objects were used as background elements, which results in them either being small in size or hidden by other objects, causing them to appear occluded. Hence, we conducted comparative experiments with the original YOLOv8n model and our proposed model by measuring mAP@0.5, mAP@0.5:0.95, and P. The results in Tables 5–7 demonstrate the efficient use of our approach. For example, the mAP@0.5, mAP@0.5:0.95, and precision for the 'Boat' category improved by 2.2%, 3.2%, and 1.7%, respectively, relative to the baseline YOLOv8n. These results indicate the effectiveness of our model in handling both occluded and small objects.

Table 5. The mAP@0.5 values corresponding to small object categories in the PASCAL VOC dataset.

Methods	Boat	Bottle	Plant	mAP@0.5
YOLOv8n	75.1%	73.7%	58.3%	69.03%
Ours	77.3%	74.4%	59.1%	70.26%

Table 6. The mAP@0.5:0.95 values corresponding to small object categories in the PASCAL VOC dataset.

Methods	Boat	Bottle	Plant	mAP@0.5:0.95
YOLOv8n	49.5%	50.7%	32.8%	44.33%
Ours	52.7%	51.5%	33.7%	45.96%

Table 7. The p values corresponding to small object categories in the PASCAL VOC dataset.

Methods	Boat	Bottle	Plant	P
YOLOv8n	75%	80.6%	73.2%	76.26%
Ours	76.7%	84%	74.5%	78.4%

The effectiveness of our approach is demonstrated in Table 8, where we compare it against multiple object detectors (two-step and one-step methods). Two-step detectors, such as Faster R-CNN and R-FCN, show respectable performance in terms of accuracy (mAP@0.5 achieving 79.5% for R-FCN). Nevertheless, these models operate in a large parametric regime (134.7M for Faster R-CNN) and are computationally expensive, which restricts them from running in time-critical environments.

Single-step detectors, such as SSD512 and YOLOv4, ensure an optimal balance between accuracy and computational speed. YOLOv4, for example, achieves a mAP@0.5 of 79% with 52.9M parameters and 120G FLOPs. However, some lighter variants, such as YOLOv4-tiny and MobileNet-SSD, sacrifice accuracy for greater efficiency, with mAP@0.5 of 59.2% and 68.0%, respectively.

Our approach outperformed all the methods compared, with a mAP@0.5 of 83.9% and a mAP@0.5:0.95 of 64.4%, surpassing even advanced versions of YOLO (e.g., YOLOv8n with 83.6% and 63.5%, respectively). Our model also maintains lightweight parameters (2.9 M).

After presenting the overall performance on the PASCAL VOC dataset, we further analyzed the independent contributions of each proposed component through an ablation study. The results are reported in Table 9.

The ablation study shown in Table 9 highlights the independent effect of each proposed component. The C2f-DCNv2 module provides the most significant improvement, achieving +0.6% on mAP@0.5:0.95 compared to the baseline, which confirms its effectiveness in enhancing feature extraction. The small object detection (SOD) layer improves recall

(+0.8%) but comes with a noticeable increase in FLOPs, while CBAM also boosts recall with only a moderate increase in complexity. GhostConv successfully reduces parameters (−0.3 M) and FLOPs (−1.0 G), although at the cost of a slight drop in precision and accuracy. When all modules are combined, the model achieves a balanced trade-off, yielding 64.4% mAP@0.5:0.95 with the lowest parameter count (2.90 M), demonstrating that our integrated design achieves both efficiency and accuracy.

Table 8. Comparative study of the object detection approaches on PASCAL VOC.

Method	mAP@0.5	mAP@0.5:0.95	Params (M)	FLOPs (G)
Two-step detector				
Faster RCNN [32]	73.2%	∼	134.7	∼
Faster RCNN [32]	76.4%	∼	∼	∼
MR-CNN [33]	78.2%	∼	∼	∼
R-FCN [34]	79.5%	∼	50.9	∼
CoupleNet [35]	82.7%	∼	∼	∼
Boosting R-CNN [36]	81.9%	∼	∼	∼
Boosting R-CNN * [36]	83.0%	∼	∼	∼
One-step detector				
DSOD300 [37]	77.7%	∼	14.8	∼
SSD512 [38]	76.8%	∼	∼	∼
STDN513 [39]	80.9%	∼	∼	∼
RefineDet512 [40]	81.8%	∼	∼	∼
DSSD513 [41]	81.5%	∼	∼	∼
FERNet [42]	81.0%	∼	∼	∼
DES512 [43]	81.7%	∼	∼	∼
DFPR512 [41]	81.1%	∼	∼	∼
EFIPNet512 [44]	81.8%	∼	∼	∼
REFBNNet512 [45]	82.2%	∼	∼	∼
SqueezeNet-SSD [46]	64.3%	∼	5.5	1.18
MobileNet-SSD [46]	68.0%	∼	5.5	1.14
Pelee [46]	70.9%	∼	5.98	1.21
Tiny-DSOD [46]	72.1%	∼	1.0	1.06
BiFPNet [47]	73.4%	58.2%	10.0	10.5
YOLOv4 [48]	79.0%	57.3%	52.9	120
YOLOv4-tiny [49]	59.2%	28.5%	5.9	16.2
YOLOv5s [50]	76.5%	50.2%	7.1	16
YOLOv5m [50]	81.5%	57.5%	20.9	48.3
YOLOv7-tiny [51]	72.7%	47.5%	6.1	13.3
YOLO-SK [50]	79.1%	55.0%	6.9	16.1
Dynamic YOLO [52]	83.3%	∼	8.27	12.56
YOLOv8n	83.6%	63.5%	3.2	8.9
Ours	83.9%	64.4%	2.9	11.9

Note: For several two-step and earlier one-step detectors, parameters and FLOPs are not reported in their original publications. These values are marked as "∼". To ensure a fair comparison, we emphasize evaluation against recent lightweight detectors with complete metrics. * means that the model uses the second training recipe.

Table 9. Ablation study of each proposed component on YOLOv8n.

Model	P (%)	R (%)	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Params (M)	FLOPs (G)
YOLOv8n (Baseline)	82.4	75.9	83.6	63.5	3.20	8.9
+Small Object Detecting Layer	82.1	76.7	83.8	63.9	2.92	12.2
+C2f-DCNv2	83.3	76.3	84.1	64.1	3.62	8.7
+CBAM	81.8	76.8	83.8	63.8	3.23	9.0
+GhostConv	82.9	74.7	83.0	63.2	2.91	7.9
Full (All modules)	82.5	76.7	83.9	64.4	2.90	11.9

4.5. Generalization Results on KITTI Dataset

To assess the generalization of the proposed model outside the PASCAL VOC dataset, supplementary experiments were conducted using the KITTI benchmark. KITTI records more complex and dynamic real-world driving scenarios. Table 10 compares the main performance metrics between the proposed method and the baseline model.

Table 10 summarizes the comparative results between our model and the original YOLOv8n model on the KITTI dataset using three major indicators: mAP@0.5, mAP@0.5:0.95, and Params. We observe that YOLOv8n achieves a mAP@0.5 of 91.8% and a mAP@0.5:0.95 of 69.6% with 3.2 million parameters. Our model surpasses YOLOv8n by achieving a mAP@0.5 of 92.6%, representing a gain of 0.8%, and a mAP@0.5:0.95 of 70.9%, corresponding to an improvement of 1.3%. In addition, our model is more compact, reducing the number of parameters by 0.3 million (2.9 M vs. 3.2 M). The increase in performance, especially at the mAP@0.5:0.95 level, demonstrates the overall generalization capacity and robustness of our model across a wider range of IoU thresholds. This is particularly important in autonomous driving scenarios, where the detection of small, occluded, and dynamic objects with high precision is crucial.

Table 10. Comparative analysis of the detection performance on the KITTI dataset.

Methods	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Params (M)
YOLOv8n	91.8	69.6	3.2
Ours	92.6	70.9	2.9

5. Conclusions

Due to the evolution of neural network-driven methods, their positive impact on object detection is becoming increasingly evident. However, existing object detection models face challenges when dealing with complex scenes, like detecting small objects, occluded objects, and multiscale objects. Additionally, these models require high computational power, which limits their usage on embedded systems. We propose a lightweight algorithm based on YOLOv8n as a solution for these issues.

To strengthen the model's capability to accurately detect small targets, a dedicated detection head is introduced, expanding the total number of detection layers to four. Additionally, the C2f layer in the backbone is replaced with the C2f-DCNv2 module to enhance feature extraction. The module relies on deformable convolution (DCNv2) to efficiently capture features from objects with complex appearances and shapes. Thirdly, we designed the CBAM lightweight attention module to address missed detections by integrating it into the neck. This incorporation involves exploiting both the spatial attention and channel attention modules, which can be effective in extracting feature information flow across the model, thereby enhancing the capacity to understand and interpret complex scenes. Ultimately, our goal is to develop an effective method that minimizes the total number of parameters in our improved approach. GhostConv, as a lightweight convolutional layer, is used to alternate with ordinary convolution in the neck, ensuring good detection performance.

Tests on the PASCAL VOC dataset and the KITTI dataset indicate that our model yields higher detection accuracy while maintaining a low number of parameters. Especially in complex scenes, our model demonstrates clear advantages compared to the baseline YOLOv8n and other existing methods. This confirms that the proposed modifications improve small, occluded, and multiscale object detection while preserving real-time performance. It also opens novel possibilities for applications requiring accurate and efficient detection in complex scenarios.

Future research will lead in several significant directions. Firstly, we aim to adapt the model to a broader range of real-world scenarios, such as object detection under various

weather conditions. To further enhance its robustness, future research will investigate multimodal data fusion by integrating information from images, LiDAR, and radar sensors. Another important direction is to combine the model with supporting perception modules. These include semantic segmentation for scene understanding at depth and natural language processing (NLP) to facilitate interaction through voice or text input. Adding NLP would enable the capability to interpret voice commands and generate descriptive feedback, facilitating collaboration between the vision and language modules. Overall, these improvements will build a stronger and more intelligent perception system, which will enhance the performance of our method on small object, occluded object, and multiscale object detection tasks. Ultimately, this work will facilitate autonomous driving and other computer vision applications.

Author Contributions: Conceptualization, S.E.H., B.H. and S.E.F.; methodology, S.E.H.; validation, B.H. and S.E.F.; formal analysis, S.E.H., B.H. and S.E.F.; investigation, S.E.H., B.H. and S.E.F.; resources, S.E.H., B.H. and S.E.F.; data curation, S.E.H.; writing—original draft preparation, S.E.H.; writing—review and editing, S.E.H., B.H. and S.E.F.; visualization, B.H. and S.E.F.; supervision, B.H. and S.E.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used in this study are publicly available. The PASCAL VOC 2007 and 2012 datasets can be accessed at https://docs.ultralytics.com/fr/datasets/detect/voc/ and the KITTI dataset at https://www.cvlibs.net/datasets/kitti/, all accessed on 25 September 2025.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ang, G.J.N.; Goil, A.K.; Chan, H.; Lee, X.C.; Mustaffa, R.B.A.; Jason, T.; Woon, Z.T.; Shen, B. A novel application for real-time arrhythmia detection using YOLOv8. *arXiv* **2023**, arXiv:2305.16727.
2. Salinas-Medina, A.; Neme, A. Enhancing Hospital Efficiency Through Web-Deployed Object Detection: A YOLOv8-Based Approach for Automating Healthcare Operations. In Proceedings of the 2023 IEEE Mexican International Conference on Computer Science (ENC), Guanajuato, Mexico, 11–13 September 2023; pp. 1–6.
3. Shi, Z. Object detection models and research directions. In Proceedings of the 2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE), Guangzhou, China, 15–17 January 2021; pp. 546–550.
4. Li, J.; Xu, R.; Ma, J.; Zou, Q.; Ma, J.; Yu, H. Domain adaptive object detection for autonomous driving under foggy weather. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 2–7 January 2023; pp. 612–622.
5. Mao, J.; Shi, S.; Wang, X.; Li, H. 3d object detection for autonomous driving: A review and new outlooks. *arXiv* **2022**, arXiv:2206.09474.
6. Chen, G.; Wang, H.; Chen, K.; Li, Z.; Song, Z.; Liu, Y.; Chen, W.; Knoll, A. A survey of the four pillars for small object detection: Multiscale representation, contextual information, super-resolution, and region proposal. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *52*, 936–953. [CrossRef]
7. 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, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
8. Zhai, S.; Shang, D.; Wang, S.; Dong, S. DF-SSD: An improved SSD object detection algorithm based on DenseNet and feature fusion. *IEEE Access* **2020**, *8*, 24344–24357. [CrossRef]
9. Xie, X.; Cheng, G.; Wang, J.; Yao, X.; Han, J. Oriented R-CNN for object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 3520–3529.
10. Pramanik, A.; Pal, S.K.; Maiti, J.; Mitra, P. Granulated RCNN and multi-class deep sort for multi-object detection and tracking. *IEEE Trans. Emerg. Top. Comput. Intell.* **2021**, *6*, 171–181. [CrossRef]
11. Giri, K.J. SO-YOLOv8: A novel deep learning-based approach for small object detection with YOLO beyond COCO. *Expert Syst. Appl.* **2025**, *280*, 127447.

12. Li, M.; Chen, Y.; Zhang, T.; Huang, W. TA-YOLO: A lightweight small object detection model based on multi-dimensional trans-attention module for remote sensing images. *Complex Intell. Syst.* **2024**, *10*, 5459–5473. [CrossRef]
13. Ma, S.; Lu, H.; Liu, J.; Zhu, Y.; Sang, P. Layn: Lightweight multi-scale attention yolov8 network for small object detection. *IEEE Access* **2024**, *12*, 29294–29307. [CrossRef]
14. Chen, Y.; Yuan, X.; Wang, J.; Wu, R.; Li, X.; Hou, Q.; Cheng, M.M. YOLO-MS: Rethinking multi-scale representation learning for real-time object detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2025**, *47*, 4240–4252. [CrossRef]
15. Qian, C.; Qian, J.; Wang, C.; Ye, X.; Zhong, C. A Vision Enhancement and Feature Fusion Multiscale Detection Network. *Neural Process. Lett.* **2024**, *56*, 19. [CrossRef]
16. Zhong, R.; Peng, E.; Li, Z.; Ai, Q.; Han, T.; Tang, Y. SPD-YOLOv8: An small-size object detection model of UAV imagery in complex scene. *J. Supercomput.* **2024**, *80*, 17021–17041. [CrossRef]
17. Luo, Q.; Wu, C.; Wu, G.; Li, W. A Small Target Strawberry Recognition Method Based on Improved YOLOv8n Model. *IEEE Access* **2024**, *12*, 14987–14995. [CrossRef]
18. Li, X.; Fu, C.; Li, X.; Wang, Z. Improved faster R-CNN for multi-scale object detection. *J. Comput.-Aided Des. Comput. Graph.* **2019**, *31*, 1095–1101. [CrossRef]
19. Terven, J.; Córdova-Esparza, D.M.; Romero-González, J.A. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1680–1716. [CrossRef]
20. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 764–773.
21. 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, Long Beach, CA, USA, 15–20 June 2019; pp. 9308–9316.
22. Jacobsen, J.H.; Van Gemert, J.; Lou, Z.; Smeulders, A.W. Structured receptive fields in cnns. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2610–2619.
23. Li, L.; Li, B.; Zhou, H. Lightweight multi-scale network for small object detection. *PeerJ Comput. Sci.* **2022**, *8*, e1145. [CrossRef] [PubMed]
24. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
25. Chen, Y.; Zhang, X.; Chen, W.; Li, Y.; Wang, J. Research on recognition of fly species based on improved RetinaNet and CBAM. *IEEE Access* **2020**, *8*, 102907–102919. [CrossRef]
26. He, K.; Sun, J. Convolutional neural networks at constrained time cost. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5353–5360.
27. 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, Seattle, WA, USA, 13–19 June 2020; pp. 1580–1589.
28. Yang, X.; Ji, W.; Zhang, S.; Song, Y.; He, L.; Xue, H. Lightweight real-time lane detection algorithm based on ghost convolution and self batch normalization. *J. Real-Time Image Process.* **2023**, *20*, 69. [CrossRef]
29. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
30. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The kitti vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
31. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
32. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
33. Gidaris, S.; Komodakis, N. Object detection via a multi-region and semantic segmentation-aware cnn model. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1134–1142.
34. Dai, J.; Li, Y.; He, K.; Sun, J. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2016; Volume 29.
35. Zhu, Y.; Zhao, C.; Wang, J.; Zhao, X.; Wu, Y.; Lu, H. Couplenet: Coupling global structure with local parts for object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 4126–4134.
36. Song, P.; Li, P.; Dai, L.; Wang, T.; Chen, Z. Boosting R-CNN: Reweighting R-CNN samples by RPN's error for underwater object detection. *Neurocomputing* **2023**, *530*, 150–164. [CrossRef]
37. Shen, Z.; Liu, Z.; Li, J.; Jiang, Y.G.; Chen, Y.; Xue, X. Dsod: Learning deeply supervised object detectors from scratch. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1919–1927.
38. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part I 14; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.

39. Zhou, P.; Ni, B.; Geng, C.; Hu, J.; Xu, Y. Scale-transferrable object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 528–537.
40. Zhang, S.; Wen, L.; Bian, X.; Lei, Z.; Li, S.Z. Single-shot refinement neural network for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4203–4212.
41. Kong, T.; Sun, F.; Tan, C.; Liu, H.; Huang, W. Deep feature pyramid reconfiguration for object detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 169–185.
42. Fan, B.; Chen, W.; Cong, Y.; Tian, J. Dual refinement underwater object detection network. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part XX 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 275–291.
43. Zhang, Z.; Qiao, S.; Xie, C.; Shen, W.; Wang, B.; Yuille, A.L. Single-shot object detection with enriched semantics. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 5813–5821.
44. Pang, Y.; Wang, T.; Anwer, R.M.; Khan, F.S.; Shao, L. Efficient featurized image pyramid network for single shot detector. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7336–7344.
45. Liu, S.; Huang, D. Receptive field block net for accurate and fast object detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 385–400.
46. Li, Y.; Li, J.; Lin, W.; Li, J. Tiny-DSOD: Lightweight object detection for resource-restricted usages. *arXiv* **2018**, arXiv:1807.11013.
47. Zhao, J.; Zhu, H.; Niu, L. Bitnet: A lightweight object detection network for real-time classroom behavior recognition with transformer and bi-directional pyramid network. *J. King Saud Univ.-Comput. Inf. Sci.* **2023**, *35*, 101670. [CrossRef]
48. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934. [CrossRef]
49. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. Scaled-yolov4: Scaling cross stage partial network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13029–13038.
50. Wang, S.; Hao, X. YOLO-SK: A lightweight multiscale object detection algorithm. *Heliyon* **2024**, *10*, e24143. [CrossRef]
51. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
52. Chen, J.; Er, M.J. Dynamic YOLO for small underwater object detection. *Artif. Intell. Rev.* **2024**, *57*, 165. [CrossRef]