Mathematical Modelling of Engineering Problems
Vol. 11, No. 11, November, 2024, pp. 3071-3081
Journal homepage: http://iieta.org/journals/mmep

Adaptive Compression Techniques for Lightweight Object Detection in Edge Devices

Nguyen Duc Toan, Long Ho Le, Hoanh Nguyen*

Faculty of Electrical Engineering Technology, Industrial University of Ho Chi Minh City, Ho Chi Minh City 700000, Vietnam

Corresponding Author Email: nguyenhoanh@iuh.edu.vn

https://doi.org/10.18280/mmep.111119

Received: 1 August 2024
Revised: 30 September 2024
Accepted: 5 October 2024
Available online: 29 November 2024

Keywords:
lightweight object detection, edge devices, ghost convolution, spatial pyramid pooling

ABSTRACT

In this paper, we propose a novel lightweight object detection model tailored for edge devices, built upon the YOLOv5 architecture. Our model introduces three key innovations to enhance both efficiency and accuracy. First, we present an Improved Spatial Pyramid Pooling Fast (SPPF) layer that combines 1×1 convolutions with dilated convolutions to expand the receptive field while minimizing computational costs, thereby improving multi-scale feature extraction. Second, we refine the model's neck by integrating Ghost and Partial Convolutions (PGhostNetV2), significantly reducing the computational load while preserving fine-grained spatial details essential for accurate detection. Finally, we enhance the Cross Stage Partial (CSP) Bottleneck by incorporating Ghost and Shuffle Convolutions (GSConv), optimizing feature representation while maintaining a lightweight structure. These enhancements result in a model that achieves competitive performance in terms of detection accuracy while significantly reducing inference time and computational demands, making it highly suitable for real-time applications on resource-constrained edge devices.

1. INTRODUCTION

Object detection stands as a critical task in the field of computer vision, with wide-ranging applications from autonomous driving to surveillance systems [1-6]. The advent of deep learning has significantly propelled the field of object detection, leading to substantial improvements in both accuracy and speed of detection. Advances in neural network architectures and learning algorithms have opened up new possibilities for robust detection mechanisms that can handle complex visual data in various conditions.

Object detection methodologies based on deep learning are generally categorized into two main types: one-stage and two-stage approaches. One-stage detectors, such as YOLO (You Only Look Once) [7-12], SSD (Single Shot MultiBox Detector) [13], FCOS [14], and CenterNet [15] are designed for speed and efficiency, processing the entire image in a single pass to predict both bounding boxes and class probabilities. Two-stage detectors, like R-CNN [16] and its variants (Fast R-CNN, Faster R-CNN) [17, 18], focus on achieving higher accuracy by first proposing candidate object regions and then classifying each region into object categories.

Among the numerous models developed for object detection, the YOLO family has established itself as a dominant framework for real-time detection due to its high-speed performance and strong accuracy. YOLOv5, in particular, has achieved a balance between precision and speed, making it a widely adopted model for a variety of real-time applications. However, there remain challenges in deploying such models on resource-constrained environments, such as edge devices, where computational and memory efficiency are paramount. This presents a need for further refinements to enhance both the efficiency and accuracy of object detection models specifically optimized for these environments.

In this work, we address these challenges by proposing a novel lightweight object detection model that builds upon the YOLOv5 architecture with specific enhancements aimed at improving performance in resource-constrained environments, such as edge devices. The novelty of our approach lies in three key architectural innovations that significantly improve the model's efficiency while maintaining competitive detection accuracy:

- Improved Spatial Pyramid Pooling Fast (SPPF) Layer: We introduce an enhanced SPPF layer that combines 1×1 convolutions with dilated convolutions, thereby expanding the receptive field without increasing computational overhead. This novel configuration enhances multi-scale feature extraction and contributes to faster inference.
- Refined Neck Architecture with Ghost and Partial Convolutions (PGhostNetV2): Our model incorporates a refined neck design that integrates Ghost and Partial Convolutions to drastically reduce the computational burden while preserving the fine-grained spatial details essential for accurate object detection. This allows the model to operate efficiently on devices with limited processing power.
- Enhanced Cross Stage Partial (CSP) Bottleneck with GSConv: We further optimize the Cross Stage Partial (CSP) layer by replacing standard convolutional operations with Ghost and Shuffle Convolutions

(GSConv). This enhancement improves feature extraction and representation while keeping the computational demands minimal.

These innovations make significant advances over the state-of-the-art by improving the balance between speed and accuracy in real-time object detection on edge devices. The proposed model is designed to be highly adaptable to resource-constrained environments, providing a practical solution for applications such as autonomous driving, surveillance, and mobile devices where computational efficiency is critical. Our experimental results, conducted on the BDD100K dataset, demonstrate that the proposed model not only achieves competitive accuracy but also significantly reduces inference time and computational demands, making it a state-of-the-art solution for lightweight object detection.

2. RELATED WORK

Object detection has seen significant advancements over the past decade, primarily driven by the rise of deep learning and convolutional neural networks (CNNs). Early approaches such as R-CNN, Fast R-CNN, and Faster R-CNN [16-18] focused on a two-stage process to achieve high accuracy. However, their computational complexity made them unsuitable for real-time applications on resource-constrained devices. This led to the development of one-stage detectors like YOLO [7-12], SSD [13], FCOS [14], and CenterNet [15], which balance speed and accuracy by predicting bounding boxes and class probabilities in a single pass. Among these, YOLO has gained widespread use for its efficiency and high detection speed.

Recently, several lightweight object detection models specifically optimized for edge devices have emerged to address the limitations of traditional models. MobileNet [19] and its variants, including MobileNetV2 [20] and MobileNetV3 [21], introduced depthwise separable convolutions to reduce computational cost, making them highly suitable for mobile and embedded systems. However, these models sometimes suffer from reduced accuracy due to the simplified network structure. EfficientDet [22] leveraged compound scaling to balance the network's depth, width, and resolution, allowing for scalable detection across different device capacities. Although EfficientDet achieved strong results in terms of accuracy and speed, its architecture is more complex and can become computationally expensive for very resource-limited devices. Another prominent lightweight approach is GhostNet [23], which introduced Ghost Convolutions to generate more feature maps from fewer operations, reducing redundancy in CNNs. GhostNetV2 [24] further refined this by improving the efficiency of feature extraction. However, while GhostNet reduces computation, its use of depthwise convolutions limits its ability to capture global context, making it less effective in scenarios where fine-grained spatial information is crucial for accurate object detection.

Another approach to enhance efficiency involves improving specific components within existing architectures. For example, the Spatial Pyramid Pooling (SPP) layer, introduced in early CNN models [25], has been adapted and optimized in various forms, including in YOLOv4 and YOLOv5, to improve multi-scale feature extraction. The introduction of techniques like dilated convolutions has further helped in capturing contextual information without increasing the computational cost [26].

Our approach builds upon the strengths of these models while addressing their limitations. Specifically, we incorporate Partial Convolutions (PConv) into GhostNetV2 to form PGhostNetV2, which overcomes the limitations of depthwise convolutions by better preserving spatial information through a split-and-convolve approach. This refinement not only reduces the computational load but also improves detection accuracy, particularly for small objects and complex environments. Further, techniques such as the Improved SPPF layer in our model are designed to improve multi-scale feature extraction, a feature shared by EfficientDet but achieved more efficiently in our architecture through the combination of dilated convolutions and 1×1 convolutions. Moreover, our Enhanced CSP Bottleneck with GSConv further optimizes feature extraction, reducing complexity without sacrificing detection performance, particularly in real-time applications on edge devices.

3. METHODOLOGY

3.1 YOLOv5 structure

The YOLOv5 architecture is widely recognized for its balance between detection accuracy and computational efficiency, making it a popular choice for real-time object detection tasks. YOLOv5 is part of the broader YOLO family, known for pioneering a one-stage detection approach where the model predicts both bounding boxes and class probabilities directly from the full image in a single pass. This design significantly reduces inference time compared to two-stage models such as R-CNN, Fast R-CNN, and Faster R-CNN, where region proposals are generated first and classified later.

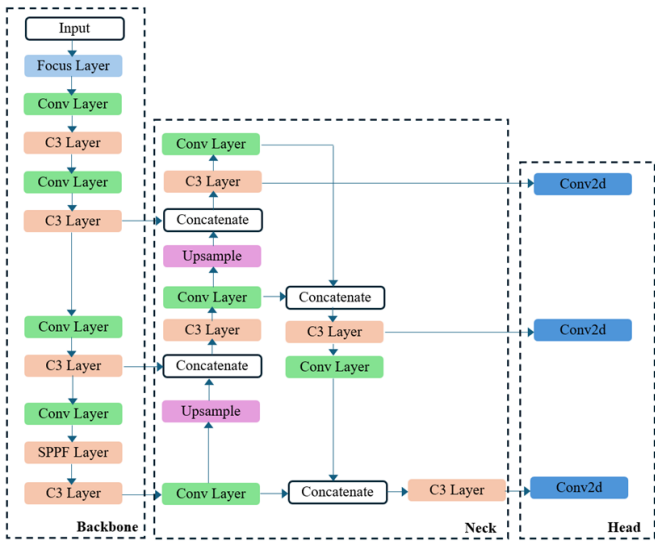

Figure 1. YOLOv5 architecture

The YOLOv5 architecture, as depicted in Figure 1, is structured into three main components: the backbone, the neck, and the head. The backbone of YOLOv5 begins with an Input layer where the input image is fed into the network. The first notable layer is the Focus layer, which helps in reducing the input dimensionality and concentrates on the most informative parts of the image for detection. This is followed by a series of Convolutional (Conv) layers and C3 (Cross Stage Partial networks) layers. The Conv layers are standard convolutional

layers used for feature extraction. The C3 layers, which are a modified version of the residual networks, enable the flow of information and gradients through the network, helping in learning more complex patterns without increasing the computational cost. The architecture also incorporates a Spatial Pyramid Pooling-Fast (SPPF) layer towards the end of the backbone. The SPPF layer pools features at different scales and concatenates them to maintain spatial hierarchies between features, enhancing the network's ability to recognize objects at various scales. The neck of YOLOv5 features a series of additional convolutional layers and upsampling layers combined with concatenation operations. This structure is crucial for constructing a rich feature pyramid which is beneficial for detecting objects at different scales. The use of upsampling layers and concatenations helps in merging the low-level feature information from earlier layers with high-level features from deeper layers, which enhances the feature representation for different object sizes. The C3 layers in the neck further help in refining these features, ensuring that the features are robust and contain useful spatial and contextual information. The head of the network consists mainly of Conv2d layers. These layers are tasked with converting the rich, multi-scale feature maps produced by the backbone and neck into the final output predictions. The head processes these features to produce the bounding boxes, object class probabilities, and objectness scores which indicate the presence of objects within the boxes.

In this work, YOLOv5 was selected as the base architecture due to its inherent advantages over other detection models. Its architecture is both computationally efficient and highly accurate, making it a prime candidate for adaptation in edge computing environments where processing power and memory are limited. Additionally, YOLOv5's modularity allows for straightforward integration of improvements such as our proposed enhancements to the Spatial Pyramid Pooling, neck structure, and Cross Stage Partial Bottleneck, which further optimize the model for lightweight detection on edge devices.

Building upon YOLOv5, we introduce several modifications designed to enhance both efficiency and detection accuracy. These include the Improved SPPF layer that expands the receptive field while reducing computational overhead, the PGhostNetV2 architecture in the neck that reduces the model's computational complexity, and the Enhanced CSP Bottleneck with GSConv for more efficient feature extraction. Together, these innovations build on YOLOv5's strengths and push the boundaries of lightweight, real-time object detection in resource-constrained environments.

3.2 Improved SPPF layer

In YOLOv5, the SPPF layer is an adaptation of the traditional Spatial Pyramid Pooling (SPP) layer [25], designed to be more efficient while maintaining similar benefits. The SPPF layer in YOLOv5 aims to enhance the receptive field of the network, allowing it to handle input of varying sizes and capture contextual information at different scales more efficiently. The detailed structure of the SPPF layer is shown in Figure 2(a). The SPPF layer utilizes a single max pooling operation with a large kernel size (i.e., 5×5) and performs max pooling with different strides to aggregate spatial features at varying scales efficiently. Mathematically, this can be expressed as:

$$P_k(x) = MaxPool(x, kernel\_size = 5 \times 5, stride = s_k) \quad (1)$$

where, $P_k(x)$ represents the max pooling operation with stride s_k, applied to the input x. The outputs of these pooling operations are concatenated along the channel dimension:

$$y = Concat(P_1(x), P_2(x), P_3(x), \dots) \quad (2)$$

This concatenated output maintains a rich representation of both local and more global features, effectively combining them to enhance the network's ability to detect objects at different scales. The main advantage of the SPPF layer over the traditional SPP is its efficiency. By using fewer pooling operations and leveraging varying strides, the SPPF layer reduces the computational overhead while still enhancing the receptive field. This makes it particularly suitable for real-time applications where both speed and accuracy are crucial.

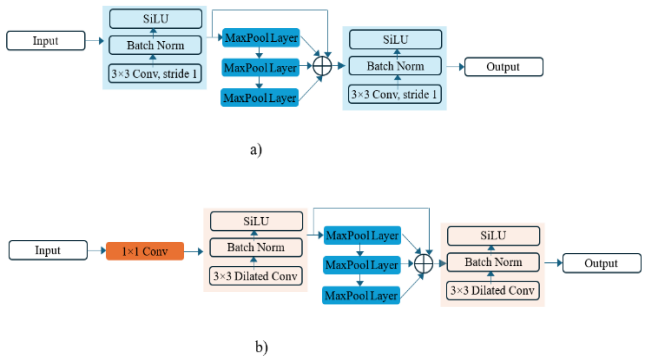

Figure 2. The structure of SPPF (a) and improved SPPF (b)

To optimize inference speed while aiming to maintain accuracy, we propose an improved SPPF layer which integrates a 1×1 convolution layer and 3×3 dilated convolutions, as shown in Figure 2(b). The Improved SPPF layer in our model is a significant enhancement aimed at capturing multi-scale context more efficiently, particularly for real-time object detection on resource-constrained edge devices. This improvement is based on integrating 1×1 convolutions with dilated convolutions to optimize both feature extraction and computational efficiency. The 1×1 convolution acts as a bottleneck layer that effectively reduces the number of input channels (i.e., feature maps):

$$y_1 = Conv_{1\times1}(x) \quad (3)$$

This operation plays a critical role in decreasing the overall computational cost by minimizing the number of parameters that need to be processed in the subsequent layers. Despite reducing dimensionality, the 1×1 convolution preserves the essential features, ensuring that the most relevant information is passed through the network. By acting as a bottleneck, the 1×1 convolution prepares the feature maps for more efficient processing through dilated convolutions.

After the 1×1 convolution, the resulting feature maps are processed by dilated convolutions (3×3 convolutions with varying dilation rates):

$$y_2 = DilatedConv_{3\times3}(y_1, dilation = d) \quad (4)$$

where, the dilation rate d increases the receptive field without

increasing the kernel size.

Dilated convolutions are effective in expanding the receptive field without increasing the kernel size, meaning the network can capture more contextual information from the input data while keeping the number of parameters low. This is particularly beneficial for object detection tasks where understanding the spatial relationships between objects at different scales is crucial. The dilation rate controls the spacing between the kernel weights, allowing the network to gather information from a broader area of the image. By combining multiple dilated convolutions with varying dilation rates, the SPPF layer can aggregate features at multiple scales, ensuring that both local and global context is captured without increasing the computational load. This multi-scale feature extraction is crucial for detecting objects of varying sizes within the same image.

This approach allows the network to capture more contextual and spatial information from the input, which is crucial for tasks like object detection where understanding the broader scene context improves accuracy. The sequence includes multiple max pooling layers, each designed to aggregate features from varying spatial extents:

$$P_k(y_2) = MaxPool \\ (y_2, kernel\_size = 5 \times 5, stride = s_k) \tag{5}$$

The outputs of these pooling operations are then concatenated:

$$y = Concat(P_1(y_2), P_2(y_2), P_3(y_2), \dots) \tag{6}$$

Supporting these layers are SiLU activation functions and batch normalization, which promote better gradient flow and faster model convergence by standardizing the activations from the convolutional layers. This structured approach enhances inference speed without compromising the model's ability to perform accurately, making it particularly effective for real-time applications where computational resources are limited. Through these improvements, the architecture is optimized for performance on diverse hardware platforms, balancing the demands of speed and precision adeptly.

3.3 Improve the neck by ghost and partial convolutions

GhostNetV2 [24] is an evolution of the GhostNet architecture [23], designed to enhance the efficiency of convolutional neural networks by reducing the redundancy present in feature maps. In GhostNetV2, as depicted in Figure 3, the basic building block is the Ghost module. This module generates more feature maps from the intrinsic features of the original feature maps, which allows for a reduction in the number of convolutional operations required. Specifically, the Ghost module applies depthwise convolution (DWConv) to create a large number of feature maps with minimal computational overhead. This structure enhances the model's efficiency, making it particularly suitable for deployment on edge devices with limited computational resources.

Despite its advantages, the use of DWConv in the GhostNetV2 structure has some limitations. DWConv is known for its ability to reduce computational complexity compared to standard convolutions, but it also has inherent drawbacks. One of the key limitations is that DWConv can struggle with capturing global context and may be less effective in scenarios where fine-grained spatial information is crucial. This limitation can lead to a degradation in the quality of feature representations, particularly in the context of complex object detection tasks where capturing intricate details is necessary for accurate predictions.

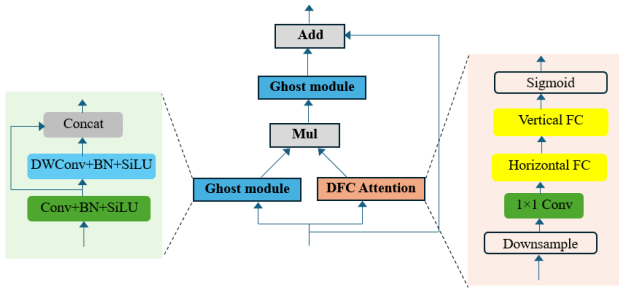

Figure 3. The architecture of GhostNetV2

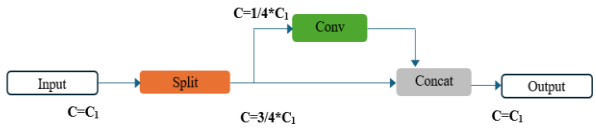

Figure 4. Partial convolution structure

To address the limitations of DWConv, we introduce Partial Convolution (PConv) into the GhostNetV2 architecture to create PGhostNetV2 module. As illustrated in Figure 4, PConv works by splitting the input feature map into two parts. One part undergoes standard convolution operations, while the other part remains untouched. This design balances the computational savings of depthwise convolutions with the spatial feature-preserving qualities of standard convolutions. Specifically, partial convolutions apply convolution only to a subset of the input feature channels while the remaining channels pass through unchanged. This selective processing allows the network to retain critical spatial information across the feature map, which is essential for accurate object detection, particularly for small objects or in scenarios where object boundaries are complex. In contrast to depthwise convolutions, where each feature map is processed independently, partial convolutions enable the model to retain both local and global spatial dependencies by processing a portion of the feature maps through regular convolutions. This approach ensures that important spatial details are preserved without a significant increase in computational cost.

The primary motivation for replacing DWConv with PConv in the GhostNetV2 structure is to overcome the aforementioned limitations of DWConv while maintaining the lightweight nature of the model. PConv, with its split-and-convolve approach, allows the model to capture more detailed spatial information without significantly increasing the computational cost. This is particularly beneficial for object detection tasks, where precise localization and recognition of objects are critical. By integrating PConv into the GhostNetV2 architecture to generate PGhostNetV2, we aim to enhance the quality of feature maps generated by the Ghost module, leading to better performance in object detection tasks, especially in edge device scenarios.

We use three PGhostNetV2 modules at the end of the neck to generate output feature maps, ensuring that they are both rich in detail and computationally efficient. The use of PGhostNetV2 modules at the end of the neck enhances the

model's ability to produce detailed and efficient feature maps, leading to improved performance on lightweight object detection tasks.

3.4 Enhanced C3 layer

The C3 layer is a critical component in YOLOv5's architecture, designed to enhance the learning capability and efficiency of the model. This layer incorporates several key features that optimize both the computational cost and the effectiveness of the neural network in processing spatial features for object detection tasks. The structure of the C3 layer begins with splitting the input feature map into two separate paths, as shown in Figure 5(a). One path processes the features directly, typically through a series of convolutional operations, while the other path is subjected to a sequence of transformations intended to refine and enhance the feature representation. The transformations generally include multiple convolutional layers; the first layer often uses a smaller number of filters to reduce dimensionality and computational load, followed by a batch normalization layer and an activation function like Leaky ReLU or SiLU to introduce non-linearity and stabilize the network learning. After processing through these convolutional blocks, the two paths are recombined. This recombination is a distinctive aspect of the C3 layer, where the feature maps from both paths are concatenated along the channel axis. This concatenation helps in enriching the feature space with both processed and bypassed features, enhancing the model's ability to capture and utilize more complex patterns and dependencies within the data. Moreover, the C3 layer includes skip connections, similar to those used in residual networks, which help in mitigating the vanishing gradient problem by allowing gradients to flow directly through the network layers during backpropagation. This is particularly beneficial for the training of deep networks, ensuring more stable and faster convergence.

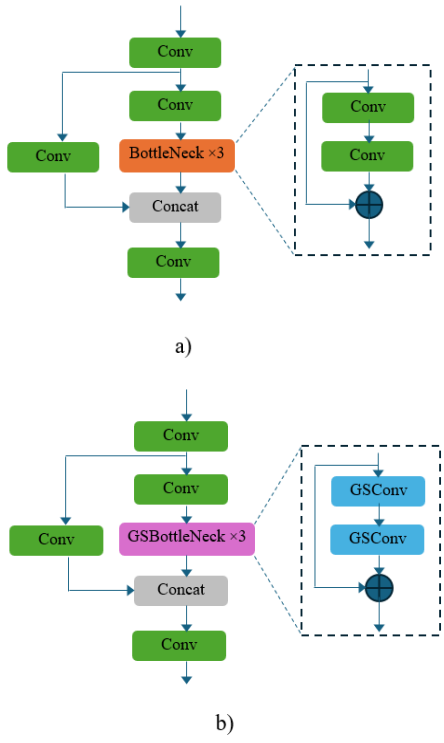

Figure 5. The structure of C3 layer (a) and enhanced C3 layer (b)

To optimize processing and improve the network's capability to handle complex object detection tasks, we design an Enhanced C3 layer, which is used to replace the standard C3 layer in YOLOv5 architecture, specifically aimed at increasing the model's efficiency and effectiveness in feature extraction and representation. The core innovation in this Enhanced C3 layer lies in the replacement of standard convolutional operations in the bottleneck structure with GSConv (Ghost and Shuffle Convolution) [27], as shown in Figure 5(b). Ghost Convolutions are designed to reduce the redundancy in feature maps by generating a portion of the output feature maps through cheaper transformations. Rather than applying full convolutions to every feature map, Ghost Convolutions apply standard convolutions to only a portion of the input features, generating the remaining feature maps through inexpensive linear operations. This drastically reduces the number of operations required, leading to a more efficient convolutional layer. The intuition behind this is that many feature maps in deep convolutional networks contain a high degree of redundancy, and Ghost Convolutions eliminate this inefficiency by learning fewer but more meaningful feature maps. Shuffle Convolutions complement Ghost Convolutions by introducing an additional mechanism that enhances the network's ability to mix and propagate information across feature channels. Shuffle Convolutions achieve this by dividing the input feature maps into groups, performing grouped convolutions on these subsets, and then shuffling the output channels to improve interaction between feature groups. This channel shuffling increases the diversity of features and allows the model to capture a wider range of spatial dependencies across the input image. The intuition here is that simply reducing the computational load, as Ghost Convolutions do, can sometimes limit the representational power of the network. Shuffle Convolutions counteract this limitation by ensuring that information from different feature maps is mixed and spread across the network, increasing the ability of the model to represent complex patterns and fine details.

The GSConv approach segments the convolutional processing into groups, allowing the network to reduce computational overhead significantly while still capturing spatial dependencies effectively. Each GSConv operates on a subset of input features, thereby focusing on extracting more nuanced and localized feature representations which can be crucial for tasks requiring high-detail perception, such as in autonomous vehicle navigation or intricate object detection scenarios. Moreover, the GS-Bottleneck modules are applied repeatedly, providing a deepened layering strategy to progressively refine features at different levels of abstraction. The bottleneck modules, now powered by GSConv, ensure that each stage of feature processing builds on the refined outputs of the previous stage, enhancing the network's ability to learn complex patterns with greater accuracy. The inclusion of concatenation after the GS-Bottlenecks and before the final convolution layer further enriches the feature maps by combining diverse representations, ensuring that subsequent layers have access to a broad spectrum of features from both deep and shallow levels. This design is aimed at boosting the representational power of the network without a commensurate increase in computational demands.

By combining Ghost and Shuffle Convolutions, the Enhanced C3 Layer achieves a balance between computational efficiency and the ability to capture detailed spatial information. Ghost Convolutions reduce the overall number of

operations without sacrificing meaningful feature representation, while Shuffle Convolutions ensure that the information in these features is mixed effectively, improving the network's ability to capture complex relationships between objects in the image. This synergy allows the Enhanced C3 layer to retain the lightweight nature required for edge devices while significantly improving the model's capacity for object detection, especially in complex or crowded environments.

4. RESULTS AND DISCUSSION

4.1 Dataset and evaluation metrics

For the evaluation of our proposed model, we utilized the BDD100K dataset [28], which is one of the largest and most diverse datasets available for autonomous driving tasks. Created by the Berkeley DeepDrive Center, this dataset is specifically designed for real-world driving scenarios and provides a rich collection of images with various driving conditions, environments, and object types, making it ideal for benchmarking object detection models for applications in autonomous driving. The BDD100K dataset contains 100,000 images, all captured at 720p resolution from dash-mounted cameras in vehicles. These images are annotated with a wide variety of objects relevant to driving scenarios. The dataset includes annotations for 10 object categories, which cover both traffic-related and environmental objects essential for autonomous vehicle navigation. In addition to object annotations, BDD100K also provides labels for various scene attributes, including weather conditions (clear, rainy, foggy, overcast), time of day (daytime, night, dusk), and road types (city streets, highways, residential areas). This diverse and comprehensive labeling makes the dataset particularly challenging and valuable for developing and testing object detection models in dynamic driving environments.

We selected the BDD100K dataset for evaluation due to several factors. First, the dataset covers a wide range of scenarios that an autonomous vehicle may encounter in real-world settings, including different lighting conditions, weather types, and varying traffic densities. This ensures that the trained model can handle complex situations, making it more robust and generalizable to real-world applications. Second, BDD100K provides images from real driving environments, which include not only urban settings but also highways, residential areas, and parking lots. This diversity in driving conditions allows us to evaluate the model's ability to detect objects accurately across different road types and environments. Finally, the BDD100K dataset includes both densely populated urban scenes with numerous small objects and sparser highway environments, challenging the model's ability to detect objects at different scales and in varying densities. Additionally, the presence of diverse weather conditions and lighting environments tests the robustness of the model across different visual conditions.

We use several metrics to assess different aspects of the model's efficiency and accuracy, including mAP, Params, FLOPs, and FPS. These metrics provide valuable insights into the model's performance. mAP is a standard metric for measuring the accuracy of object detectors like those used in computer vision. It represents the average precision across all classes and recall levels. Precision is the ratio of correctly predicted positive observations to the total predicted positives, and recall is the ratio of correctly predicted positive observations to all actual positives. The mAP is calculated by taking the mean of the Average Precision (AP) for each class. Params (Parameters) refers to the total count of trainable parameters in the model. It is a direct indicator of the model's complexity and memory requirements. Models with a higher number of parameters might be capable of learning more detailed features but are also generally more computationally intensive and slower to train. FLOPs measure the computational complexity of the model, specifically the number of floating-point operations required to generate an output from a single input. This metric is crucial for understanding the computational demand of the model, particularly in deployment scenarios where processing power is a limiting factor. FPS measures the speed of the model in processing input frames. It is an essential metric for applications requiring real-time processing, such as video analysis and autonomous driving. Higher FPS indicates that the model can process more frames in a shorter amount of time, enhancing the responsiveness of the application.

4.2 Experimental setup

In this section, we provide a detailed description of the experimental setup used to train and evaluate the proposed model. This information is crucial for reproducibility and clarity regarding the model's performance.

For all experiments, we used an input resolution of 640×640 pixels. This resolution was chosen based on the balance between computational efficiency and detection accuracy, especially for real-time object detection tasks on edge devices. By using a fixed input resolution, we ensured consistency across all experiments and maintained the model's real-time performance. We initialized our model using weights pre-trained on the COCO dataset. Pre-training on COCO provides a strong starting point for object detection tasks as the dataset contains a wide variety of objects across many different categories, which helps the model learn basic features such as edges, textures, and object shapes.

For the experimental evaluation of our proposed model, we utilized a hardware setup consisting of an Intel Core i7-11700K CPU and an NVIDIA RTX 4080 GPU. This combination provides a robust platform for high-performance computation, necessary for handling the intensive demands of real-time object detection tasks. The model was trained using a batch size of 8 and an initial learning rate of 0.001, adjusted dynamically with a cosine annealing schedule to optimize convergence. We employed standard data augmentation techniques such as random cropping, rotation, and color adjustment to enhance the model's robustness to real-world variations in input data. Training was conducted over 100 epochs, with early stopping implemented to prevent overfitting based on the validation loss.

4.3 Comparison with other models

We compared the performance of our model with various models on the BDD100K dataset, including MultiNet [29], Faster R-CNN [18], YOLOV5s, DLT-Net [30], YOLOP [31], and HybridNets [32]. The comparison results are presented in Table 1. The results demonstrate that our proposed model, which incorporates significant enhancements over the standard YOLOv5 architecture, outperforms most existing models in terms of both accuracy and speed. With a Recall of 90.6% and an mAP50 of 77.8%, our model not only achieves

higher precision in object detection compared to models like MultiNet, Faster R-CNN, DLT-Net, and YOLOP but also surpasses the original YOLOv5s and is slightly more accurate than HybridNets. Crucially, it maintains a high detection accuracy while significantly boosting the inference speed to 84 FPS, which is markedly higher than all compared models. This high performance can be attributed to the architectural improvements in our model. The Improved SPPF layer is designed to optimize inference speed without compromising accuracy, effectively balancing computational efficiency with robust detection capabilities. The refined neck architecture using Ghost and Partial Convolutions reduces the computational load while maintaining high-quality feature extraction. Additionally, the introduction of the Enhanced C3 layer, replacing the standard C3 layer in YOLOv5, contributes to more effective feature representation and extraction, further boosting the model's overall performance. These enhancements allow our model not only to perform exceptionally in terms of standard metrics but also to operate at a high speed, making it ideal for real-time applications such as autonomous driving. The significant improvement in FPS, in particular, underscores the model's suitability for deployment in scenarios where quick processing of visual data is critical. The blend of high recall and superior mAP50 also indicates that the model effectively minimizes false negatives and accurately identifies objects, which is paramount in scenarios that demand high reliability, such as in varying lighting and weather conditions encountered in autonomous vehicle navigation. Thus, our model stands out as a highly efficient and effective solution in the landscape of object detection technologies, particularly for applications in dynamic and challenging environments.

To further demonstrate the generalizability of our approach beyond autonomous driving scenarios, we extended our evaluation to an additional widely-used object detection dataset: MS COCO [33]. This dataset is commonly used benchmark in the computer vision community and provide diverse images with a wide range of object categories, environments, and scene complexities. The MS COCO dataset contains over 330,000 images and 80 object categories, with challenging annotations that include both large and small objects in cluttered scenes. This dataset is ideal for testing the generalization of our model because it covers a wide variety of contexts that go beyond the autonomous driving domain. The COCO dataset also includes annotations for segmentation and keypoint detection, making it a comprehensive benchmark for real-world applications. Table 2 provides comparison results. The results in Table 2 show a clear trade-off between speed and accuracy among the various models. YOLOX-Tiny [34] and MobileNet-SSD [35] excel in terms of speed, achieving 150 FPS and 160 FPS, respectively, but have lower accuracy compared to our model. Our model strikes an optimal balance, with an mAP50 of 48.5% and a competitive speed of 145 FPS, outperforming YOLOv5s in both accuracy and speed. While Faster R-CNN lags in speed at just 10 FPS, it also has the lowest mAP50, indicating that it is less suited for real-time edge deployment compared to other models. These results on MS COCO demonstrate the generalizability of the proposed model to a broader set of object detection tasks, validating its effectiveness across different domains beyond autonomous driving.

When developing lightweight object detection models, one of the most critical factors is the trade-off between inference speed and accuracy. Models designed for edge deployment must be computationally efficient to ensure real-time processing while maintaining a high level of detection accuracy, particularly in complex environments where missed detections can be costly. Our proposed model strikes a fine balance between inference speed and accuracy through several architectural enhancements, including the Improved SPPF layer, PGhostNetV2, and the Enhanced C3 Layer. These innovations were specifically designed to optimize both computation and accuracy, addressing the typical limitations of lightweight models that prioritize one metric at the expense of the other. For inference speed, our model achieves an impressive 145 FPS, which is competitive with highly efficient models such as MobileNet-SSD and YOLOX-Tiny. Despite the high processing speed, our model still achieves higher accuracy. This is largely due to the inclusion of Ghost Convolutions and Partial Convolutions, which reduce the number of redundant operations, thus speeding up inference without losing critical feature extraction capabilities. In addition, the use of dilated convolutions in the Improved SPPF layer enables the model to capture multi-scale features without adding significant computational overhead, contributing to both speed and accuracy.

Table 1. Comparison with other models on the BDD100K

Models	Recall	mAP50 (%)	Speed (FPS)
MultiNet [29]	81.3	60.2	18
Faster R-CNN [18]	77.2	55.6	16
YOLOV5s	86.8	77.2	66
DLT-Net [30]	89.4	68.4	25
YOLOP [31]	89.2	76.5	45
HybridNets [32]	92.8	77.3	24
Our model	90.6	77.8	84

Table 2. Comparison with other models on the MS COCO

Models	Recall	mAP50 (%)	Speed (FPS)
MultiNet [29]	65.2	38.5	30
Faster R-CNN [18]	61.0	37.5	10
YOLOV5s	68.0	44.1	140
DLT-Net [30]	69.5	41.2	55
YOLOP [31]	70.8	42.9	55
YOLOX-Tiny [34]	66.5	47.4	150
MobileNet-SSD [35]	62.4	38.0	160
Our model	71.2	48.5	145

4.4 Comparison with the baseline model

We also evaluated our model in various environments and compared the results with the baseline model, YOLOv5. The results presented in Table 3 illustrate the robust performance of our model across a variety of environmental scenarios when compared to the baseline model, YOLOv5. In urban settings such as city streets, highways, and residential areas, our model consistently outperforms YOLOv5, achieving mAP50 scores of 0.624, 0.602, and 0.645 respectively, compared to 0.572, 0.539, and 0.608 by YOLOv5. This indicates a stronger ability to accurately detect objects in complex urban environments where diverse objects and movement patterns are present. Furthermore, our model demonstrates significant improvements in more controlled environments such as tunnels, gas stations, and parking lots, with the most notable increase seen in parking lots where the mAP50 score jumps from 0.654 to 0.760. This substantial improvement suggests that our model is particularly effective in environments with structured spaces and stationary objects.

Weather conditions also pose varying challenges; however, our model shows enhanced performance under all weather scenarios. For instance, in clear conditions, it achieves an mAP50 of 0.631 compared to 0.572 by YOLOv5, and even in more challenging foggy and snowy conditions, it scores 0.600 and 0.617, respectively, outperforming the baseline. This enhanced capability under diverse weather conditions highlights the model's adaptability and reliability, crucial for applications such as autonomous driving where weather variability is a significant factor. Additionally, the model's performance during different times of the day, daytime, night, and dawn/dusk, shows consistent improvements. It scores 0.622, 0.607, and 0.621 in these conditions, respectively, compared to 0.572, 0.554, and 0.586 by YOLOv5. This improvement across various lighting conditions further confirms the model's robustness in variable lighting, enhancing its utility in real-world applications where lighting can change unpredictably. Overall, the enhancements in our model have clearly contributed to its superior performance across different scenarios, making it a more versatile and reliable option for real-time object detection in diverse environments and conditions.

Table 3. Performance results in various driving scenarios

Scenarios	Environments	mAP50	
		YOLOv5	Our Model
Scenes	City street	0.572	0.624
	Highway	0.539	0.602
	Residential	0.608	0.645
	Tunnel	0.641	0.674
	Gas stations	0.638	0.653
	Parking lot	0.654	0.760
Weather	Clear	0.572	0.631
	Overcast	0.578	0.617
	Rainy	0.607	0.632
	Foggy	0.576	0.600
	Snowy	0.548	0.617
	Undefined	0.559	0.603
Time	Daytime	0.572	0.622
	Night	0.554	0.607
	Dawn/dusk	0.586	0.621

Table 4. The effect of each component

Baseline	Improved SPPF Layer	PGhostNetV2 in the Neck	Enhanced C3 Layer	mAP50(%)	Params	GFLOPs	FPS
√				77.2	7 M	16.4	66
√	√			77.3	6.2 M	15.2	68
√		√		77.7	7.2 M	16.8	59
√			√	77.4	6.4 M	14.8	73
√	√	√	√	77.8	5.6 M	14.3	84

4.5 Ablation study

We conduct ablation experiments to evaluate the effect of each component proposed in this paper. Table 4 presents the ablation results. The ablation study presented in Table 4 systematically evaluates the impact of various architectural enhancements on the performance of our model. This approach allows us to discern the individual and combined contributions of each proposed component: Improved SPPF layer, the refined neck architecture with PGhostNetV2 modules, and Enhanced C3 layers. Starting with the baseline model, which achieves an mAP50 of 77.2%, we observe incremental improvements as each component is introduced. The introduction of the Improved SPPF layer slightly increases the mAP50 to 77.3%, while simultaneously reducing the model parameters to 6.2 million and GFLOPs to 15.2, thus enhancing both model efficiency and slight performance. This indicates that the Improved SPPF layer contributes to a more computationally efficient architecture without significantly impacting accuracy. Integrating PGhostNetV2 into the neck results in a slight improvement in detection accuracy, with the mAP50 increasing from 77.2% to 77.7%. This enhancement comes with a modest increase in the number of parameters (from 7M to 7.2M) and GFLOPs (from 16.4 to 16.8), reflecting a trade-off between computational complexity and accuracy. However, this integration also leads to a reduction in FPS from 66 to 59, indicating a decrease in inference speed due to the added computational demands.

The addition of the Enhanced C3 layer to the baseline leads to the improvement in mAP50, reaching 77.4%, with a notable reduction in GFLOPs to 14.8 and parameters to 6.4 million, alongside an increase in FPS to 73. This demonstrates that the Enhanced C3 layer significantly contributes to both the efficiency and effectiveness of the model, optimizing feature extraction and representation capabilities more substantially than the other components. When all enhancements are combined, the model achieves an mAP50 of 77.8%, which is the highest score among the configurations. Additionally, this full configuration yields the most significant reductions in GFLOPs (14.3) and model parameters (5.6 million), while also achieving the highest FPS (84). These results highlight the synergistic effect of integrating all proposed improvements, leading to a model that not only performs better in terms of accuracy but is also substantially more efficient and faster. This combination makes it particularly suited for real-time applications where both performance and computational speed are critical.

While our proposed lightweight object detection model demonstrates strong performance in terms of accuracy and efficiency, several practical deployment challenges must be considered when implementing the model on various edge devices. One of the most significant challenges when deploying deep learning models on edge devices is memory limitation. Although we have minimized the number of parameters in the model (5.6M parameters compared to YOLOv5s with 7M), further reductions might be necessary depending on the edge device. For instance, devices like microcontrollers or small embedded systems may not have enough memory to load the entire model, especially when handling large batches or higher-resolution inputs. Techniques such as model quantization, where weights are reduced from 32-bit floats to 8-bit integers, can further reduce memory usage without significant loss of accuracy. Another option is model pruning, where unnecessary weights and neurons are removed post-training to reduce the model size. Both approaches can help address memory constraints during deployment on memory-limited edge devices. Another critical factor in edge deployment is the balance between power

consumption and latency. Many edge devices, particularly those operating in remote or resource-constrained environments, are battery-powered and have strict requirements for real-time performance. To address latency concerns, edge-cloud hybrid deployment models can be used, where more computationally intensive tasks are offloaded to the cloud, while real-time, lightweight tasks are handled locally on the edge device.

4.6 Visualization results

Figure 6 provides visualization results of our model in various environmental and lighting conditions. Across different scenarios, city streets, residential areas, gas stations, tunnels, highways, and parking lots, the model consistently demonstrates high accuracy in detecting vehicles, showcasing the effectiveness of the Improved SPPF, PGhostNetV2, and Enhanced C3 layers. In complex urban settings like city streets and residential areas, where multiple vehicles and pedestrians coexist, the model accurately identifies smaller and partially occluded objects, a testament to its refined feature extraction capabilities. In more controlled environments such as tunnels and parking lots, where lighting and background conditions are more uniform, the model performs exceptionally well, highlighting its capacity to maintain high detection standards even in less variable contexts. This is particularly notable in parking lots during dawn or dusk, where the model successfully delineates vehicles despite the challenging light conditions, indicating strong performance in low-light scenarios.

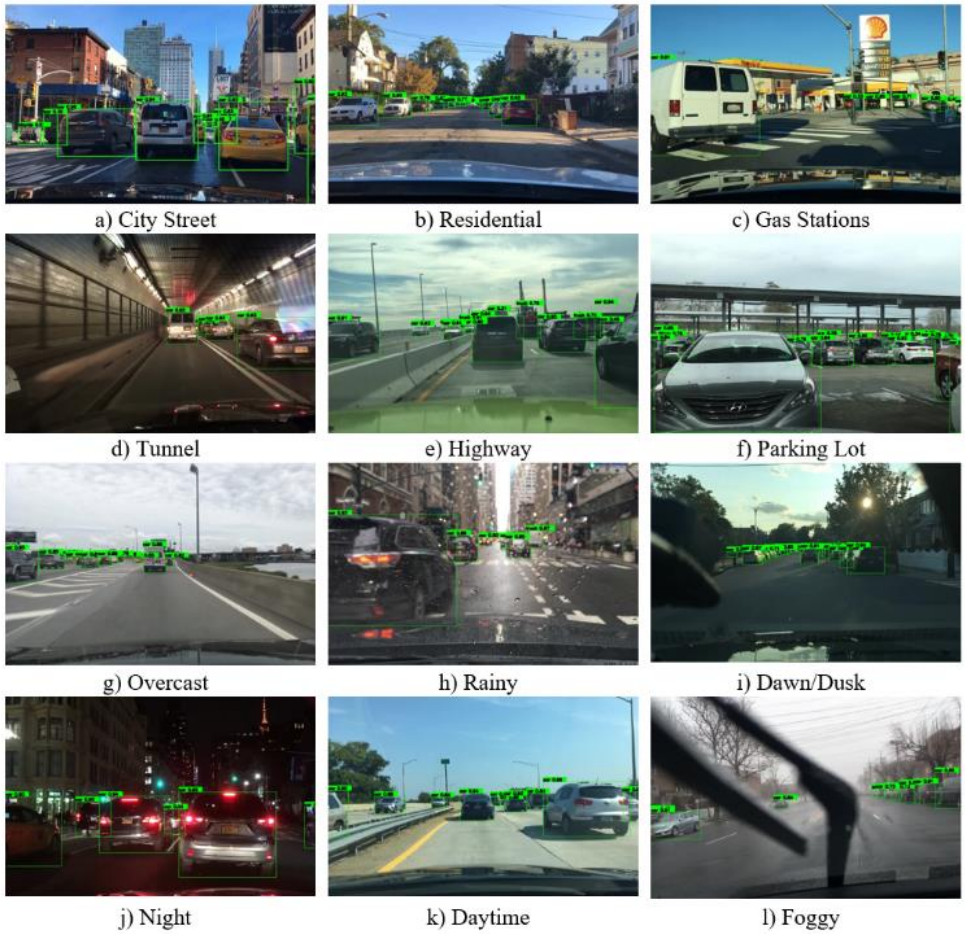

Figure 6. Visualization results of our model in different scenarios

Weather conditions, often a challenge for object detection systems, are adeptly handled by the model. In rain, fog, and overcast conditions, the model retains high detection accuracy. The clarity of object detection in rainy conditions underscores the model's resilience to visual distortions caused by water on the lens or other reflective surfaces. Furthermore, during night-time conditions, our model effectively utilizes available lighting, such as street lamps and vehicle headlights, to ensure reliable detection, which is critical for applications like autonomous driving that require 24/7 operational capabilities. Moreover, the distinct bounding boxes and labels visible in the images indicate precise localization and classification of each object, which are essential for real-time decision-making systems in autonomous vehicles. The enhanced feature extraction capabilities of the model, enabled by the architectural improvements, ensure that each vehicle is accurately tracked across different frames and conditions, facilitating reliable and safe navigational decisions.

While our proposed lightweight object detection model demonstrates strong performance across various datasets, like all models, it is prone to certain failure modes. One of the most common failure modes is related to the detection of small objects. In scenarios where objects like pedestrians, traffic lights, or distant vehicles occupy only a few pixels in the image, the model sometimes struggles to accurately detect or classify them. This issue becomes more pronounced in cluttered scenes where multiple small objects overlap. One potential solution is to further enhance the multi-scale feature

extraction capabilities of the model. While the current use of dilated convolutions in the Improved SPPF layer helps capture multi-scale context, additional fine-tuning could be done to increase the model's sensitivity to smaller objects. Occlusion is another common failure mode where the model struggles to detect objects that are partially obstructed. For example, in autonomous driving scenarios, vehicles and pedestrians are often occluded by other objects like trees, signs, or other vehicles. In such cases, the model might miss detecting the partially visible objects. Improving the model's ability to handle occlusions can be addressed by enhancing its contextual reasoning capabilities. For instance, the inclusion of attention mechanisms or feature pyramid networks with enhanced context-awareness could help the model understand the global context better and predict partially visible objects. In low-light conditions or under adverse weather (e.g., rain, fog), the model tends to generate false positives due to the lack of clear object boundaries. These false positives typically involve background elements being incorrectly classified as objects. One solution is to use domain-specific data augmentation, such as adding synthetic fog, rain, or low-light conditions to the training data to improve robustness in such scenarios.

The analysis of these failure modes highlights the model's strong performance in most scenarios but also points to areas where improvements can be made. By addressing the challenges of small object detection, occlusion handling, and false positives in adverse conditions, the robustness of the model can be further improved.

5. CONCLUSIONS

In this paper, we presented a series of enhancements to the YOLOv5 architecture, specifically aimed at improving the performance of object detection models on edge devices with limited computational resources. Our proposed model incorporates three key innovations: an Improved SPPF layer to expand the receptive field while maintaining computational efficiency, a refined neck architecture using Ghost and Partial Convolutions (PGhostNetV2) to reduce the computational load while preserving detailed spatial feature extraction, and an Enhanced C3 layer that utilizes GSConv for optimized feature representation. Through extensive experimentation on the BDD100K and MS COCO datasets, we demonstrated that our model achieves competitive performance in terms of accuracy while significantly reducing inference time and computational demands. These results highlight the potential of our approach for real-time applications, particularly in environments where processing power and energy efficiency are critical, such as autonomous vehicles, surveillance systems, and other edge computing scenarios.

An important aspect of the proposed architectural changes is their generalizability. The enhancements introduced in this paper, such as Ghost Convolutions, Partial Convolutions, and the multi-scale feature extraction capability of the Improved SPPF layer, are not limited to YOLOv5 and can be applied to other backbone networks. For instance, these improvements can benefit architectures such as MobileNet, EfficientDet, or even more complex backbones like ResNet or DenseNet when used in lightweight or real-time detection scenarios. By integrating these modifications, other models could achieve similar reductions in computational overhead while maintaining or even improving detection accuracy. The modular nature of these enhancements makes them adaptable to a variety of network structures. Therefore, future work could explore applying the proposed improvements to alternative backbones, with the expectation that they would yield similar benefits in terms of inference speed and computational efficiency, particularly on resource-constrained devices.

REFERENCES

[1] Boukabous, M., Azizi, M. (2023). Image and video-based crime prediction using object detection and deep learning. Bulletin of Electrical Engineering and Informatics, 12(3): 1630-1638. https://doi.org/10.11591/eei.v12i3.5157

[2] Arrahmah, A.I., Rahmania, R., Saputra, D.E. (2022). Comparison between convolutional neural network and K-nearest neighbours object detection for autonomous drone. Bulletin of Electrical Engineering and Informatics, 11(4): 2303-2312. https://doi.org/10.11591/eei.v11i4.3784

[3] Widodo, C.E., Adi, K., Priyono, P., Setiawan, A. (2023). An evaluation of pre-trained convolutional neural network models for the detection of COVID-19 and pneumonia from chest X-ray imagery. Mathematical Modelling of Engineering Problems, 10(6): 2210-2216. https://doi.org/10.18280/mmep.100635

[4] Patil, H., Bhosale, S. (2023). Enhancing few-shot learning for tropical cyclone severity prediction: A deep learning approach. Mathematical Modelling of Engineering Problems, 10(6): 2239-2248. https://doi.org/10.18280/mmep.100639

[5] Wen, H., Song, K., Huang, L., Wang, H., Yan, Y. (2023). Cross-modality salient object detection network with universality and anti-interference. Knowledge-Based Systems, 264: 110322. https://doi.org/10.1016/j.knosys.2023.110322

[6] Yang, L., Zhang, X., Li, J., Wang, L., Zhu, M., Zhu, L. (2023). Lite-FPN for keypoint-based monocular 3D object detection. Knowledge-Based Systems, 271: 110517. https://doi.org/10.1016/j.knosys.2023.110517

[7] Joseph, R., Divvala, S., Girshick, R., Farhadi, A. (2016). 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. pp. 779-788. http://doi.org/10.1109/CVPR.2016.91

[8] Redmon, J., Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, pp. 7263-7271. http://doi.org/10.1109/CVPR.2017.690

[9] Redmon, J. (2018). Yolov3: An incremental improvement. arxiv preprint arxiv:1804.02767. https://doi.org/10.48550/arXiv.1804.02767

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

[11] Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., Wei, X. (2022). YOLOv6: A single-stage object detection framework for industrial applications. arxiv preprint arxiv:2209.02976.

https://doi.org/10.48550/arXiv.2209.02976
[12] Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M. (2023). 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, pp. 7464-7475. https://doi.org/10.1109/CVPR52729.2023.00721
[13] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C. (2016). SSD: Single shot multibox detector. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, pp. 21-37. http://doi.org/10.1007/978-3-319-46448-0_2
[14] Tian, Z., Shen, C., Chen, H., He, T. (1904). FCOS: Fully convolutional one-stage object detection. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), pp. 9626-9635. https://doi.org/10.1109/ICCV.2019.00972
[15] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q. (2019). Centernet: Keypoint triplets for object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea (South), pp. 6569-6578. https://doi.org/10.1109/ICCV.2019.00667
[16] Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, pp. 580-587. https://doi.org/10.1109/CVPR.2014.81
[17] Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, pp. 1440-1448. https://doi.org/10.1109/ICCV.2015.169
[18] Ren, S., He, K., Girshick, R., Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6): 1137-1149. http://doi.org/10.1109/TPAMI.2016.2577031
[19] Howard, A.G. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. arxiv preprint arxiv:1704.04861. https://doi.org/10.48550/arXiv.1704.04861
[20] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, pp. 4510-4520. https://doi.org/10.1109/CVPR.2018.00474
[21] Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q.V., Adam, H. (2019). Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea (South), pp. 1314-1324. https://doi.org/10.1109/ICCV.2019.00140
[22] Tan, M., Pang, R., Le, Q.V. (2020). Efficientdet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, pp. 10781-10790. https://doi.org/10.1109/CVPR42600.2020.01079
[23] Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C. (2020). GhostNet: More features from cheap operations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, pp. 1580-1589. https://doi.org/10.1109/CVPR42600.2020.00165
[24] Tang, Y., Han, K., Guo, J., Xu, C., Xu, C., Wang, Y. (2022). GhostNetv2: Enhance cheap operation with long-range attention. Advances in Neural Information Processing Systems, 35: 9969-9982. https://doi.org/10.48550/arXiv.2211.12905
[25] He, K., Zhang, X., Ren, S., Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(9): 1904-1916. https://doi.org/10.1109/TPAMI.2015.2389824
[26] Yu, F., Koltun, V., Funkhouser, T. (2017). Dilated residual networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, pp. 472-480. https://doi.org/10.1109/CVPR.2017.75
[27] Li, H., Li, J., Wei, H., Liu, Z., Zhan, Z., Ren, Q. (2024). Slim-neck by GSConv: A lightweight-design for real-time detector architectures. Journal of Real-Time Image Processing, 21(3): 62. https://doi.org/10.1007/s11554-024-01436-6
[28] Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., Darrell, T. (2020). Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, pp. 2636-2645. https://doi.org/10.1109/CVPR42600.2020.00271
[29] Teichmann, M., Weber, M., Zoellner, M., Cipolla, R., Urtasun, R. (2018). Multinet: Real-time joint semantic reasoning for autonomous driving. In 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, pp. 1013-1020. https://doi.org/10.1109/IVS.2018.8500504
[30] Qian, Y., Dolan, J.M., Yang, M. (2019). DLT-Net: Joint detection of drivable areas, lane lines, and traffic objects. IEEE Transactions on Intelligent Transportation Systems, 21(11): 4670-4679. https://doi.org/10.1109/TITS.2019.2943777
[31] Wu, D., Liao, M.W., Zhang, W.T., Wang, X.G., Bai, X., Cheng, W.Q., Liu, W.Y. (2022). Yolop: You only look once for panoptic driving perception. Machine Intelligence Research, 19(6): 550-562. https://doi.org/10.1007/s11633-022-1339-y
[32] Vu, D., Ngo, B., Phan, H. (2022). HybridNets: End-to-end perception network. arxiv preprint arxiv:2203.09035. https://doi.org/10.48550/arXiv.2203.09035
[33] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L. (2014). Microsoft coco: Common objects in context. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, pp. 740-755. https://doi.org/10.1007/978-3-319-10602-1_48
[34] Zheng, G., Liu, S.T., Wang, F., Li, Z.M., Sun, J. (2021). YOLOX: Exceeding YOLO series in 2021. arXiv preprint arXiv:2107.08430. https://doi.org/10.48550/arXiv.2107.08430
[35] Chiu, Y.C., Tsai, C.Y., Ruan, M.D., Shen, G.Y., Lee, T.T. (2020). Mobilenet-SSDv2: An improved object detection model for embedded systems. In 2020 International Conference on System Science and Engineering (ICSSE), Kagawa, Japan, pp. 1-5. https://doi.org/10.1109/ICSSE50014.2020.9219319