Expert Systems With Applications 234 (2023) 121036

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

An improved lightweight small object detection framework applied to real-time autonomous driving

Bharat Mahaur *, K.K. Mishra[1], Anoj Kumar

Department of Computer Science and Engineering, Motilal Nehru National Institute of Technology Allahabad, UP, India

ARTICLE INFO

Keywords:
Small object detection
Architectural changes
YOLOv5
Kernel pruning
Lightweight design
Autonomous driving

ABSTRACT

Recent deep learning-based object detectors have shown compelling performance for the detection of large objects in autonomous driving applications. However, the detection of small objects like traffic signs and traffic lights is challenging owing to the complex nature of such objects. This article investigates how an existing object detector can be adjusted to address specific tasks and how these modifications can impact the detection of small objects. In particular, we explore and introduce architectural changes to the different components of the popular YOLOv5 model in order to improve its performance in the detection of small objects for autonomous driving. Initially, we propose group depthwise separable convolution as the improved convolution unit to replace standard convolution. We then integrate this unit to create the attention-based dilated CSP block. Lastly, this block is combined with several proposed modules, including the improved SPP, improved PANet, and improved information paths, to form our IS-YOLOv5 model. We also integrate kernel pruning on the network to accelerate the model deployment on vehicle-mounted mobile platform due to limited computing resources and real-time constraints. Specifically, we propose the versatile network pruning (VNP) technique based on Taylor criterion ranking to prune less-essential kernels in the network. We will show that our modifications barely increase the complexity but significantly improve the detection accuracy and speed. Compared to the conventional YOLOv5, the proposed IS-YOLOv5 model increases the mAP by 8.35% on the BDD100K dataset. Besides, our proposed model improves the detection speed in FPS by 3.10% compared to the YOLOv5 model. When using the VNP scheme, FPS is further increased by 52.14%, while the model size and complexity are reduced by 39.29% and 47.81%, with almost no change in mAP. Nevertheless, when compared to state-of-the-art models, IS-YOLOv5+VNP is found to be conducive to the deployment in autonomous driving systems.

1. Introduction

In the community of computer vision, object detection is a fundamental and well-studied task that aims to classify and localize target objects in an image. With the recent advancements in deep learning technologies over the years, several state-of-the-art methods for object detection have emerged (Liu, Sun, Wergeles, & Shang, 2021). Moreover, the computational speed of deep learning algorithms has been significantly accelerated by the rapid advancement of sensors, multicore processors, and GPUs. Object detection has been widely applied to many real-world applications, including autonomous driving, intelligent transportation, augmented reality, robot vision, drone imagery analysis, remote sensing, military operations, and surveillance (Badue et al., 2021; Chen, Lin, et al., 2021; Mahaur, Singh, & Mishra, 2022).

Various aspects can be used to illustrate a clear definition of small objects (Xiao et al., 2022). We refer to objects as small when their occupying pixel area or field of view is small in an input image. Several object detectors usually perform better on large and medium objects but poorly on small objects. In the case of generic object detectors, the features of small objects lose importance as they are processed through multiple layers of their backbone. The accurate detection of small objects is indispensable and challenging due to less context information, indistinguishable features, geometric cues, complicated backgrounds, limited resolution, occlusion, etc. (Badue et al., 2021; Xiao et al., 2022). Although modern systems designed to implement object detection in real-time mainly focus on speed at the cost of computational resources, they lack feasibility due to poor detection accuracy.

Detecting target objects on the road is an essential task for autonomous driving. An autonomous vehicle on the road is equipped with a variety of sensors to provide the system with accurate information

* Corresponding author.
E-mail addresses: bharatmahaur@gmail.com (B. Mahaur), kkm@mnnit.ac.in (K.K. Mishra), anojk@mnnit.ac.in (A. Kumar).
[1] Deceased

https://doi.org/10.1016/j.eswa.2023.121036
Received 23 November 2022; Received in revised form 19 July 2023; Accepted 20 July 2023
Available online 25 July 2023

about its surroundings. At present, Cameras, GPS, LiDAR, and Radar are the primary sensors used for the detection of road targets (Gupta & Mahaur, 2021; Mahaur et al., 2022). In comparison to these sensors, cameras are now more accurate and less expensive at detecting road objects in practical constraints. Although the quality of camera images is significantly impacted by adverse weather and low-light conditions, several image enhancement techniques can be developed to enhance visibility in particular scenarios. Consequently, there is a need for a fast and accurate deep learning-based road object detection system that can improve control decisions of an autonomous vehicle for real-time detection while also ensuring safety for pedestrians and other vehicles.

For most existing objector detectors, the detection accuracy for small objects is less than half that of large objects. This is because it is difficult to extract features from low resolution, and the model can easily confuse it with the background, resulting in missed or incorrect detection. Also, the detection features for small objects are insufficient as they usually cover fewer pixels and may appear in any position of the image, indicating a poor feature representation. Moreover, one of the most critical challenges of an object detector is that the accurate detection of different-sized objects is not well-balanced. In the context of autonomous driving, traffic lights and traffic signs can be regarded as small road objects. Although many studies (Chen, Wang, et al., 2020; Liu et al., 2021; Mahaur, Mishra, & Singh, 2023) suggest increasing the representational capacity of the network in terms of depth and width for accurate detection, this impacts the complexity and cost of the model. Unfortunately, such models are less suited for real-time applications because of their architecture and resource constraints. Thus, improvements in this specific area would benefit practical implications in autonomous driving systems.

In general, deep learning-based object detection models are categorized into (1) Two-stage detection algorithms and (2) One-stage detection algorithms (Zou, Shi, Guo, & Ye, 2019). The two-stage models achieve higher accuracy than one-stage models at the cost of speed and complexity but may not directly benefit the practical driving scenarios. Recently, efforts have been put into matching or even improving the performance of one-stage models (Chen, Wang, et al., 2020; Khosravian, Amirkhani, Kashiani, & Masih-Tehrani, 2021). Hence, many new one-stage detectors have been developed for such applications. In this article, we focus on the popular one-stage detector, i.e., You Only Look Once 5 (YOLOv5) model (Jocher et al., 2022). This is the most recent version in the YOLO family, with a clear and flexible structure aiming for high performance and speed on accessible platforms. Although YOLOv5 is a generic object detector, it is not optimized for the detection of small objects in autonomous driving and therefore cannot adapt to specific use cases in practice.

Many one-stage detectors have made satisfactory progress in terms of detection accuracy in past years, but they are computationally intensive and time-consuming, which restricts powerful models like RetinaNet (Lin, Goyal, Girshick, He, & Dollár, 2017) and YOLO variants (Mahaur et al., 2022) from being deployed in real-time for vehicle-mounted computing platform. Researchers have explored various approaches for accelerating such models by integrating network compression techniques (Mishra, Gupta, & Dutta, 2020). From several deep compression strategies, we choose the widely used method, i.e., network pruning. This has been proved to be an effective compression scheme for deep neural networks. The key idea of network pruning is to remove redundant components, such as layers, neurons, filters, channels, etc., in the network (Liu, Liu, Ma, Tan, & Gan, 2020; Zou et al., 2019). The compressed model consumes less power and memory resources, allowing for faster inference with minimal accuracy loss, resulting in a lightweight model.

Despite the benchmark results achieved by generic object detectors on large and medium objects, their detection accuracy on small objects is far from satisfactory. This implies that, even with state-of-the-art detection methods, there is a significant performance gap in detecting small and large objects. To address this problem, researchers have developed a few techniques like mapping dense anchors on multiple feature maps, balancing positive and negative examples through either data reweighting or loss function optimization, and providing adequate training samples (Chen, Lin, et al., 2021; Jocher et al., 2022; Xiao et al., 2022). However, such methods add extra complexity to the system and heavily depend on tailored data, which limits performance in practical scenarios. Furthermore, some technologies like combining different feature layers to enrich multi-level feature maps and incorporating attention-based mechanisms to focus more on specific features, have shown promising results (Chen, Wang, et al., 2020; Mahaur et al., 2023). More importantly, current autonomous systems that use these approaches rely either on conventional training methods, regularization/normalization techniques, or adjusting specific parameters to improve performance, with limited or no consideration for structural modifications.

Based on the aforementioned ideas, this article proposes architectural changes to the original YOLOv5 model to perform better in terms of small object detection. To be specific, we propose the improved convolution unit and create the attention-based block to extract multi-level semantic features. Also, we introduce improved structures of different components in the YOLOv5 model to provide consistent performance gains for multi-scale objects. For applicability, we consider the actual road environment in autonomous driving systems to detect small road objects like traffic signs and traffic lights. Moreover, we discuss the effects of our modifications on how to accurately perform this task while maintaining real-time speed and with a slight increase in the computational complexity of the system. At the same time, we propose the network pruning technique based on Taylor criterion ranking to prune non-essential convolution kernels for reducing model size and complexity. Redundant kernel pruning can effectively balance the sparsity of convolution kernels in different layers, thereby making the network dense and providing good generalization. Accordingly, this proposed approach will result in a lightweight model framework with competitive detection accuracy and higher speed, which is suitable for deployment on the vehicle computing platform in autonomous road driving environments. The highlights of our contributions are

- We optimize the existing YOLOv5 model and design a modified YOLOv5 architecture, with the name Improved Scaled YOLOv5 (IS-YOLOv5), aiming for better detection of small objects without compromising the detection accuracy of large objects.
- We propose the versatile network pruning (VNP) strategy based on Taylor criterion ranking to remove less-important convolution kernels in the network. This significantly reduces the size and complexity of the model while preserving the detection accuracy at faster inference.
- We construct an improved lightweight object detection framework by integrating the VNP scheme into the IS-YOLOv5 model, which promotes high applicability in the vehicle computing platform for autonomous driving systems.
- We demonstrate the efficacy of the proposed method through extensive experimentation on the BDD100K dataset. Furthermore, we report experimental results for traffic sign and traffic light detection on four challenging datasets, i.e., STSD, GTSDB, BSTLD, and S2TLD.
- We investigate the applicability of our approach in diverse weather scenarios to highlight its significance in the context of more robust and efficient object detection.

2. Related work

Many researchers have shown a significant interest in developing and applying deep learning-based models for performance enhancement in small object detection tasks. Although the increase in performance comes at the expense of higher computational costs, it has become necessary to compress and accelerate the deployment of such models on low-computing platforms. In this section, we briefly introduce the related works focused on small object detection methods and model compression techniques.

2.1. Small object detection

Over the years, some effort has been made to utilize the YOLOv5 model for the detection of small objects in autonomous driving scenarios. For instance, Benjumea, Teeti, Cuzzolin, and Bradley (2021), Chen, Jia, Chen, Lv, and Zhang (2022) and Zhu and Yan (2022) exploited the benefits of YOLOv5 for the detection of road objects like traffic signs and/or traffic lights. While such methods achieve promising results, they do not make an effort to modify the architecture. Similarly, Katsamenis et al. (2022) and Li, Song, Zhang, Liang, and Cheng (2022) implemented YOLOv5 for traffic cone detection. Both of these works attempted to optimize the YOLOv5 model, but only to a limited extent since typical changes to their original structure are common. Moreover, Bie, Liu, Li, Hong, and Li (2022) and Jiang, Zhao, Li, and Jia (2020) combined additional modules to the YOLOv5 feature extractor for refining the detection of small-scale objects. However, such methods mainly focus on inference speed, sacrificing detection accuracy at the cost of more resources, if needed. Furthermore, Li et al. (2022) and Zhu and Yan (2022) integrated commonly used attention mechanisms for developing attention-based techniques, which resulted in better detection performance. However, these approaches are not universally applicable because they do not account for increasing complexity and inference time. Besides, structural modifications have proven to be relatively more effective than other techniques for improving small object detection performance (Ge, Liu, Wang, Li, & Sun, 2021; Katsamenis et al., 2022; Mahaur & Mishra, 2023). In addition, the existing works showed detection results for small objects in plain weather conditions. Because an autonomous vehicle on the road encounters a variety of road environments, analysis made during the daytime will not suffice in challenging weather scenarios. More importantly, all of the current YOLOv5-based systems fail to measure computational cost, which is an important metric for autonomous driving. Also, in most cases (Bie et al., 2022; Chen et al., 2022), the performance improvements in small object detection tasks are triggered by parametric blocks.

Therefore, we investigate how architectural changes in the different components of the YOLOv5 model can improve detection performance at a low cost, especially for small objects. To be specific, we introduce the improved convolution unit and create the attention-based block. Furthermore, we incorporate this block with several proposed modules, including the improved SPP, improved PANet, and improved information paths. When compared to the original YOLOv5, our proposed model is able to significantly boost the detection accuracy, particularly for small objects, at a faster inference speed and with a minimal increase in computations.

2.2. Model compression

Despite the success of YOLO models in the object detection field, currently deployed systems require a significant amount of computation and storage resources to ensure high-end performance in real-time. In most cases, these requirements can directly affect the widespread deployment of detection frameworks on vehicle computing platform for autonomous driving. Fortunately, several researchers (Anwar & Sung, 2016; Cai et al., 2021; Molchanov, Tyree, Karras, Aila, & Kautz, 2016) have explored the benefits of network compression techniques in the design of lightweight network structures. Network pruning, in particular, is one of the popular methods for model compression that involves pruning inadequate network components. Few works (He, Kang, Dong, Fu, & Yang, 2018; Li, Kadav, Durdanovic, Samet, & Graf, 2016) adopted weight pruning with $L2$-norm regularization to safely prune weights that tend to absolute zero at negligible accuracy loss. However, weight pruning does not modify the network architecture and requires dedicated libraries and hardware for fast inference. Meanwhile, the works (Huang & Wang, 2020; Li et al., 2021) combined channel pruning and block pruning to achieve high compression ratios, which speeds up the real-time inference of the pruned network. Channel pruning, on the other hand, is highly dependent on the precise combination of channels to achieve a satisfying result, and an incorrect selection can result in significant accuracy loss. Unlike other pruning methods, kernel pruning can improve the performance rate without any specific requirements. Indeed, Molchanov et al. (2016) introduced a criteria-based pruning method to prune redundant neurons in the convolutional layers. Another work (Cai et al., 2021) designed a sparse scaling algorithm to regularize the scaling factor using $L1$-norm and removed the convolution kernels with a scaling factor close to zero. Both of these studies simplified computational complexity and achieved good performance in developing a lightweight model for resource-constrained platforms.

In contrast to the above research, we propose the versatile network pruning scheme based on Taylor criterion ranking to prune the least important convolution kernels from the trained network. Our pruning strategy may obtain better pruning performance compared to the conventional $L2$-norm and $L1$-norm based techniques. Essentially, this method reduces the computational cost and storage requirement of deep learning-based models while keeping almost unchanged detection performance, making them more suitable for use in autonomous vehicle systems.

3. Methodology

We first discuss the motivations of our work (Section 3.1). Second, we provide a brief overview of YOLOv5 architecture and discuss its shortcomings (Section 3.2). Third, a series of novel architectural changes are introduced to optimize and improve the detection of small objects (Section 3.3). Fourth, we present a versatile network pruning strategy based on Taylor criterion ranking to prune convolution kernels (Section 3.4). Finally, the pruning module is applied to the optimized architecture to obtain a compressed lightweight model.

3.1. Motivations

Although some techniques have been developed for enhancing the performance of small object detection, only a handful of researchers focus on architectural modifications to achieve the same. In most cases, the gain in performance is mainly driven by additional blocks or by increasing the parameters in the framework. Hence, it is unclear how architectural refinements contribute towards improving the detection performance for explicit tasks. In autonomous driving, accurate detection of small objects provides more valuable contextual information about the environment that can help better decision-making strategies. The detection of small objects is more challenging because of foreground/background imbalance, fewer appearance cues, and lower image cover rate. In a typical road traffic environment, detecting small objects is considered a hard problem as distant objects become smaller due to perspective distortion. Additionally, the complexity present in a few images induced by adverse weather conditions such as foggy, snowy, rainy, and low-light conditions makes it difficult to detect small objects accurately. Notably, there are significant differences in the localization of small objects of different sizes, even in the same class. Also, many driving systems prioritize inference time over performance, but there are workarounds to optimize them at a low cost. Consequently, a simple and more accurate road object detection model that can handle small and large objects of different sizes is needed.

Despite recent advancements in the detection performance of deep learning-based methods, the enormous amount of computations and storage requirements restrict their deployment on low-computing platform in autonomous systems. This makes it difficult for autonomous devices like self-driving vehicles to accomplish the detection task. Furthermore, the majority of existing approaches are impractical for direct use in autonomous driving environments. As a result, it is imperative to realize high detection performance on the vehicle-mounted computing platform. To compensate for this drawback, model deployment can be accelerated by applying deep compression techniques, promoting

the goal of lightweight detection. Because a trained deep network, in particular, contains a large number of non-essential components, a number of which can be pruned from the original network to reduce computational cost and storage requirement while maintaining detection performance nearly unchanged. Therefore, a lightweight and efficient object detection framework must be designed to meet the needs of autonomous driving.

Motivated by the above observations, we analyze different elements of the popular YOLOv5 architecture to improve detection performance in a specific task. To the best of our knowledge, there is no work that optimizes the structure of YOLOv5 for autonomous driving by proposing architectural changes for the accurate detection of small objects at faster inference without sacrificing the detection accuracy of large objects and with minimal increase in model complexity. Further, we dynamically compress the proposed model structure using the versatile network pruning method to minimize complexity and storage requirements while preserving the detection accuracy. These strategies accelerate the model deployment in autonomous driving scenarios by proposing an improved lightweight object detection framework.

3.2. YOLOv5

As with the most recent one-stage detection model, YOLOv5 has a strong ability for feature extraction with high detection speed and accuracy. The YOLOv5 series provides four model scales: YOLOv5 - S, YOLOv5 - M, YOLOv5 - L, and YOLOv5 - X, where S is small, M is medium, L is large, and X is xlarge (Jocher et al., 2022). The network structure of these models is constant, but the modules and convolution kernels are scaled, which alters the complexity and size of each model. In this article, we analyze and focus on the small variant of the YOLOv5 model unless otherwise specified. The basic architecture of YOLOv5 is shown in Fig. 1, including input, backbone, neck, and head.

In the input, YOLOv5 uses Mosaic data augmentation to enhance data for small-scale detection. The first part of the architecture is the backbone network, which consists of the Focus layer (Jocher et al., 2022), BottleNeckCSP (Bochkovskiy, Wang, & Liao, 2020), and SPP module (Wang, Bochkovskiy, & Liao, 2021). The primary role of the Focus layer is to perform image slicing. This reduces information loss during downsampling, simplifies numerical calculations, and boosts training speed. The BottleNeckCSP not only reduces the overall computational burden but also extracts in-depth information from the features more effectively. The SPP module is used to increase the receptive field of the network by transforming the variable-size feature map into a feature vector of fixed size. The second component is the neck network, which applies PANet (Duan et al., 2019) and FPN (Lin et al., 2017) operations. The PANet structure is used for the dense positioning of high-level features. At the same time, FPN provides powerful low-level semantic features by upsampling. Then, various multi-scale features are fused to strengthen the detection ability. The last element is the head network, which performs the final detection of different scales. It outputs a vector containing class probabilities, confidence scores, prediction angle, and bounding box information using anchor boxes.

The detection performance of YOLOv5 is very good, but we observe some practical limitations. First, it is primarily intended for generic tasks and may not be applicable to a specific task discussed in this work. Second, the standard convolution has quite a vast number of parameters and requires heavy computations, which sacrifices detection speed for real-time tasks. Third, PANet structure focuses less on information flow between non-adjacent layers, due to which the information is constantly reduced in each aggregation process. Fourth, the max-pooling operations in the SPP module result in significant information loss and thus cannot obtain global information for localization. Fifth, the convolutions in BottleNeckCSP add redundancy and complexity while having minimal effect on detection accuracy. Sixth, the information paths that connect the different components in the YOLOv5 architecture limit computational efficiency and are not optimal for extracting relevant features for small-scale objects.

3.3. Proposed architectural changes

The original YOLOv5 model needs to be improved for autonomous driving applications. We introduce several modifications to further improve detection speed and accuracy without significantly increasing the model complexity.

3.3.1. Improved standard convolution

The convolution is a fundamental operation in CNNs that uses filter kernels to generate a set of feature maps. In standard convolution (SC), each channel of the input features is convoluted by a kernel, and the number of kernels depends on the input and output channels. The number of parameters (L) and computation cost (T) of SC are calculated as

$$L_{SC} = n \cdot n \cdot C_{in} \cdot C_{out}, \tag{1}$$

$$T_{SC} = M \cdot M \cdot n \cdot n \cdot C_{in} \cdot C_{out}, \tag{2}$$

where n is convolution kernel size, M is the input image size, C_{in} and C_{out} are the numbers of input and output channels, respectively. As C_{in} and C_{out} increase for practical tasks, the multiplications will cause L and T to become quite large. Consequently, a large number of parameters will be produced by the YOLOv5 model, resulting in increased computations. To address this issue, we use depthwise separable convolution (DSC) instead of standard convolution (SC). The original DSC is a type of factorized convolution, which factorizes the SC into a depthwise convolution (DWC) and a pointwise convolution (PWC) (Sandler, Howard, Zhu, Zhmoginov, & Chen, 2018). In DWC, an individual kernel is associated with each input feature map. Then, PWC applies a convolution with a kernel size of 1×1 on the stacked input to combine the desired output. This method of factorized convolutions greatly reduces the model size and computational complexity. The L and T of DSC are computed as

$$L_{DSC} = n \cdot n \cdot C_{in} + C_{in} \cdot C_{out}, \tag{3}$$

$$T_{DSC} = M \cdot M \cdot n \cdot n \cdot C_{in} + M \cdot M \cdot C_{in} \cdot C_{out}, \tag{4}$$

The theoretical parameter ratio r_{L1} of Eq. (3) to Eq. (1) is defined as

$$\begin{aligned} r_{L1} &= \frac{n \cdot n \cdot C_{in} + C_{in} \cdot C_{out}}{n \cdot n \cdot C_{in} \cdot C_{out}} \\ &= \frac{1}{C_{out}} + \frac{1}{n^2}. \end{aligned} \tag{5}$$

Similarly, the computation ratio r_{T1} of Eq. (4) to Eq. (2) is defined as

$$\begin{aligned} r_{T1} &= \frac{M \cdot M \cdot n \cdot n \cdot C_{in} + M \cdot M \cdot C_{in} \cdot C_{out}}{M \cdot M \cdot n \cdot n \cdot C_{in} \cdot C_{out}} \\ &= \frac{1}{C_{out}} + \frac{1}{n^2}. \end{aligned} \tag{6}$$

where $C_{out} \gg 1$ and $n > 1$. As both r_{L1} and r_{T1} are less than 1, DSC has fewer parameters and computations compared to SC. Using DSC will not only reduce model size and complexity but will also shift focus to the detection of multi-scale objects. However, PWC in the second part of DSC is also SC, which increases parameters and computations. The computational cost of the model can be reduced even further if it is improved.

To compensate for this drawback, we suggest replacing the pointwise convolution in DSC with pointwise group convolution (PWGC). It is well known that group convolution (GC) can efficiently distribute the model across multiple GPUs, reducing computational cost and memory (Wu, Iandola, Jin, & Keutzer, 2017; Zhang, Zhou, Lin, & Sun, 2018). Assume the number of groups as s, then L and T of PWGC are represented as

$$L_{PWGC} = n \cdot n \cdot \frac{C_{in}}{s} \cdot \frac{C_{out}}{s} \cdot s, \tag{7}$$

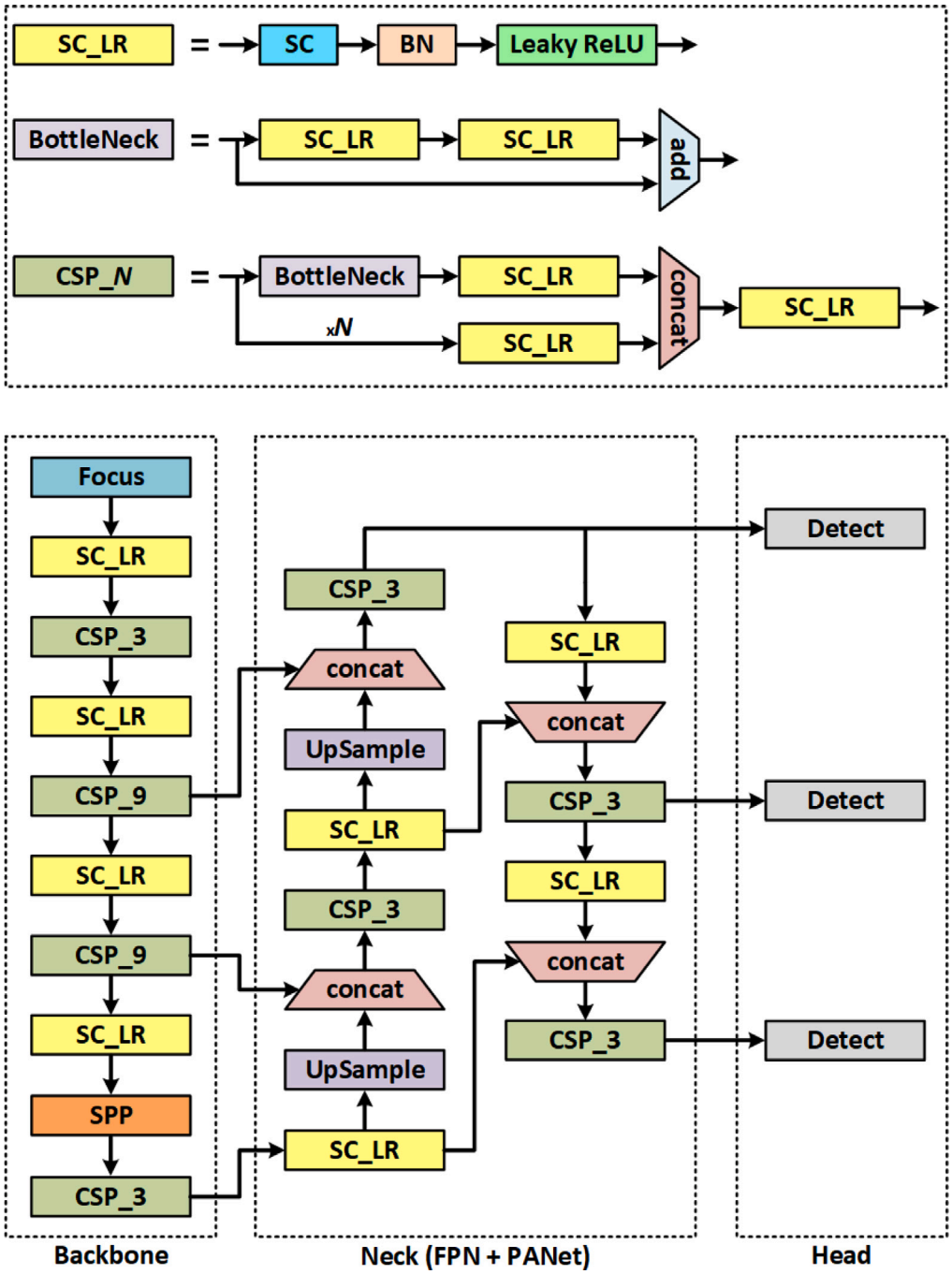

Fig. 1. Original YOLOv5 architecture.

$$T_{PWGC} = M \cdot M \cdot n \cdot n \cdot \frac{C_{in}}{s} \cdot \frac{C_{out}}{s} \cdot s, \tag{8}$$

The parameter ratio r_{L2} of Eq. (7) to Eq. (1) is stated as

$$r_{L2} = \frac{n \cdot n \cdot \frac{C_{in}}{s} \cdot \frac{C_{out}}{s} \cdot s}{n \cdot n \cdot C_{in} \cdot C_{out}} = \frac{1}{s}. \tag{9}$$

Similarly, the computation ratio r_{T2} of Eq. (8) to Eq. (2) is stated as

$$r_{T2} = \frac{M \cdot M \cdot n \cdot n \cdot \frac{C_{in}}{s} \cdot \frac{C_{out}}{s} \cdot s}{M \cdot M \cdot n \cdot n \cdot C_{in} \cdot C_{out}} = \frac{1}{s}. \tag{10}$$

where $s > 0$, then both r_{L2} and r_{L2} are less than 1. Hence, GC can acquire the same number of features with fewer parameters and computations compared to SC. In addition, GC makes a significant contribution to model compression and improves inference time.

In depthwise separable convolutions, the DWC is a special type of GC in which the number of input channels equals the number of groups, and the PWC is simply SC. Although GC minimizes the number of parameters and computations, it restricts the flow of information between groups and mitigates feature representation when multiple GC are stacked together. In other words, the outputs from a specific group are only related to the inputs from that group (Zhang et al., 2018). Hence, the channel shuffle (CS) operation is performed after the PWC to relate the input and output channels by recombining them between groups. We refer to the improved DSC as group depthwise

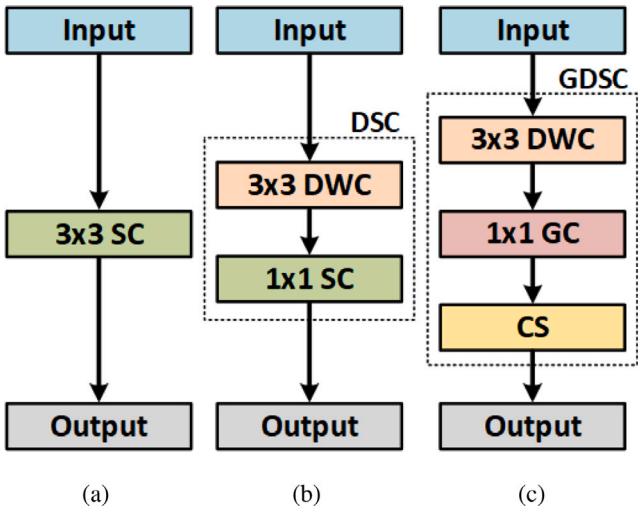

Fig. 2. (a) Standard Convolution (SC) (b) Depthwise Separable Convolution (DSC) (c) Proposed Group Depthwise Separable Convolution (GDSC).

separable convolution (GDSC), and the same is depicted in Fig. 2. The combination of DWC, GC, and CS in the GDSC minimizes the number of parameters and computing cost while rapidly adapting to the small object detection task.

3.3.2. Improved SPP module

As the depth of the CNN increase, the size of the receptive field becomes larger. Due to the limited size of input images, the feature extraction is repetitive on the large receptive fields (Niu, Hu, & Li, 2021; Wang, Bochkovskiy, & Liao, 2021). Therefore, the SPP module is used to add the corresponding modules to eliminate this problem by fusing the feature maps of different receptive fields. This module combines the local and global features to maximize the expressive ability of feature maps, extends the receiving field of the backbone network, and separates the most important context features for size target detection. For integrating the characteristics of receptive fields of different scales, the SPP uses several max-pooling operations in parallel. This method has shown benefits for improving overall detection performance. However, the max-pooling operations fail to capture the global spatial information and result in certain information loss.

To address this issue, we propose an improved SPP module by simply replacing the pooling function with dilated convolution (DC) (Chen, Wang, et al., 2021). Although both of these operations expand the network receptive field, the pooling reduces the spatial resolution, resulting in the loss of characteristic information. In contrast, the DC enriches the extracted features by capturing multi-scale information required for detecting small targets without losing resolution. By using DC, the size of the receptive field can be expanded without increasing the number of parameters and computational cost. For this purpose, the dilation rate is adjusted to produce feature maps of different sizes, which are then concatenated on the same level to enhance feature representation (Mahaur & Mishra, 2023). However, the calculation method of DC is a little tricky, so the continuous information in the input can be disrupted, which results in the loss of some features. Therefore, to avoid such a problem, we use the DC in parallel with SC to increase the network receptive field while ensuring no information loss. The receptive field F of DC is calculated as

$$F = n + (n-1)\cdot(d-1). \tag{11}$$

where n is the size of the original kernel and d is the dilation rate. We combine DC with our GDSC and refer to it as group dilated depthwise separable convolution (GDDSC), as shown in Fig. 3. Our module improves the learning ability of the network to accurately locate targets, particularly small objects while maintaining fast detection speed and high computational efficiency.

3.3.3. Improved PANet structure

The aim of the neck network is to aggregate the features obtained by the backbone for improving the accuracy of the latter prediction. This structure plays a crucial role in preventing the loss of small object information due to higher abstraction levels. To achieve this, the feature maps are upsampled once again in order to perform aggregation with backbone layers and reassert influence over the detection (Duan et al., 2019; Ge et al., 2021). The FPN provides deep semantic information to the shallow layers for enhancing detection ability but ignores the location information during the aggregation of shallow and deep features. Whereas PANet provides the fusion of both low-level and high-level information. However, the integration method relies on the aggregation of adjacent features, and less attention is given to the exchange of information among non-adjacent layers. Due to this, the spatial information in the non-adjacent layers is continuously reduced with each aggregation. Furthermore, the semantic information obtained from linear upsampling is not fully utilized for feature extraction.

Therefore, to address the above limitations, we design an improved PANet structure based on the original PANet structure, as depicted in Fig. 4. We add two cross-layer connections (B1 and B2), one in the top-down path of FPN and the other in the bottom-up path of PANet, for integrating non-adjacent and multi-level features. By applying B1 and B2, the low-level and high-level feature information can be more effectively utilized (Mahaur & Mishra, 2023). During aggregation, this will allow for more effective use of semantic and shallow location information, enhancing important features of small objects without increasing computational complexity.

3.3.4. Improved information paths

The architectural design of YOLOv5 is simple but needs optimization for computational efficiency and real-time applicability due to its internal arrangement of components. Therefore, we redirect certain connections in order to focus on detecting multi-resolution feature maps. As the input is passed layer-by-layer through the convolutions, the feature maps are extracted. The feature maps generated by the former convolutional layers capture small-scale objects, whereas the feature maps generated by the latter capture large-scale objects. The BottleNeckCSP is the most basic block of YOLOv5 and extracts most contextual features, but its current attributes are inefficient for extracting deep features (Mahaur & Mishra, 2023). As a result, the small objects receive less attention or are even ignored during the region proposal stage. Another important aspect is the selection of an appropriate activation function, which can limit performance even when adding multiple convolution and normalization layers. Moreover, the head lacks the ability to extract enough shallow features for localizing small objects.

In response to the above problems, we propose an Improved Scaled YOLOv5 (IS-YOLOv5), a robust and efficient architecture, the detailed structure of which is shown in Fig. 5. The original BottleNeckCSP block limits the inference speed of the model because the bottleneck is placed before the convolution pipeline. For justifying the limitations of the BottleNeckCSP, a new functional block is introduced called Attention-based Dilated CSP (ADCSP) by modifying the information paths and adding certain layers to improve the detection accuracy for small objects without losing speed. Specifically, we first adopt a lightweight attention mechanism, Frequency Channel Attention Networks (FCANet) (Qin, Zhang, Wu, & Li, 2021) to boost the network feature extraction ability. FCANet extends the widely known SENet to create a multi-spectral channel attention framework. This enables our network to adaptively learn weights from distinct features. Secondly, we compute the input feature maps by combining the SC and DC, then concatenate them using a shortcut connection while increasing the receptive field of the network (Mahaur & Mishra, 2023). In such a way, ADCSP can extract multi-level features with more semantic information, thus enhancing the detection accuracy of small objects.

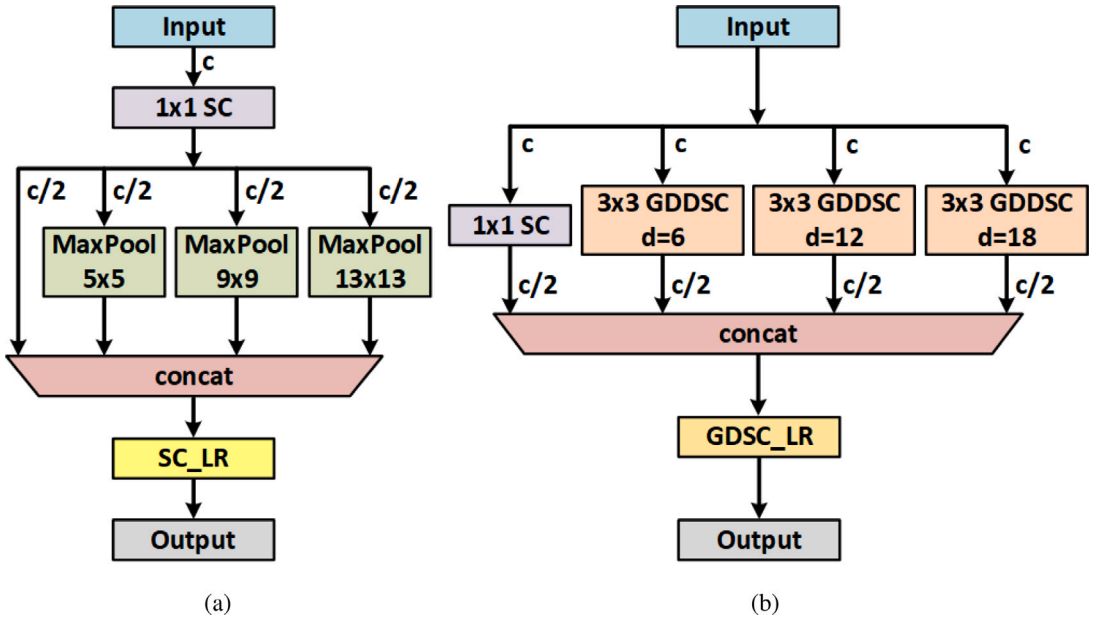

Fig. 3. (a) SPP module (b) Improved SPP module.

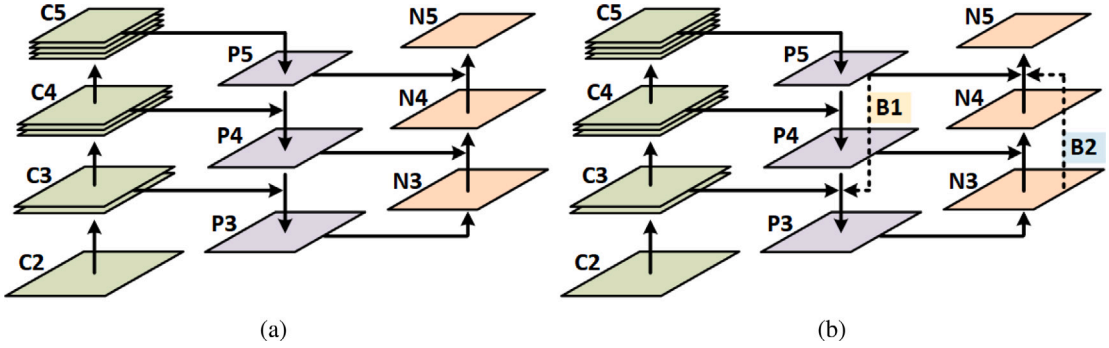

Fig. 4. (a) PANet structure (b) Improved PANet structure.

Besides, we reduce the number of ADCSP blocks in the backbone and the neck to adjust network parameters.

Furthermore, we implement Hard Swish activation instead of Leaky ReLU in specific layers of the network model. We apply multiple activations to avoid information loss and reduce computational cost according to the input size. The Hard Swish activation of z can be expressed as $z \cdot ReLU(z+3)/6$. Likewise, the Leaky ReLU of z can be specified as $1(z < 0)(\beta z) + 1(z \geq 0)(z)$, where β is a small constant. On the detection side, we add a detection head for small-scale objects obtained from high-resolution feature maps. In the neck, we optimally adjust the ADCSP blocks to focus on detecting multi-scale features. This will improve the overall detection ability for different scale objects, particularly small targets. Note that our collective modifications barely change the computational complexity while significantly improving detection performance as well as ensuring real-time requirements.

3.4. Versatile network pruning

The deployment of deep learning-based models on low-computing platforms is severely restricted due to their high computation and storage requirements. Various network pruning strategies can be applied to reduce the overall computational cost and eliminate redundancy in complex models. However, selecting an appropriate pruning algorithm is difficult as different network models have different data structures, which can lead to sensitive performance responses (Molchanov et al., 2016). Therefore, to minimize the cost and size of our network architecture without losing too much performance, we propose the versatile network pruning (VNP) scheme to prune less-essential convolution kernels of the convolutional layers.

Specifically, our VNP strategy is used to rank the convolution kernels based on their contributions. This will result in a lightweight network structure by identifying and removing the least important convolution kernels as per their relative contribution from the actual network. Based on the deployment environment of the vehicle computing platform, we specify a network pruning rate p to determine the degree of convolution kernel pruning from the original network $\mathbb{N}$. For pruning the network $\mathbb{N}$, we consider the loss function $\mathcal{L}(\mathcal{W})$ between the real and predicted outputs. We use $\mathcal{W} = \{(w_1, b_1), \ldots, (w_k, b_k), \ldots, (w_K, b_K)\}$ to describe the network parameter set, where w_k and b_k denote the weight and bias of the kth convolutional kernel f_k, and K represents the total number of kernels in the network $\mathbb{N}$. For minimizing the loss $\mathcal{L}(\mathcal{W})$ between the real and predicted output values on a dataset $\mathcal{D}$, the parameter set $\mathcal{W}$ is optimized during the training process. Essentially, with respect to identifying and pruning the less-important convolution kernels, we refine an optimal parameter set $\mathcal{W}'$ of the set $\mathcal{W}$, where a reasonable loss value is maintained as close to the original as possible during the entire process of pruning. This can be formulated as a convolution kernel-based combinatorial optimization

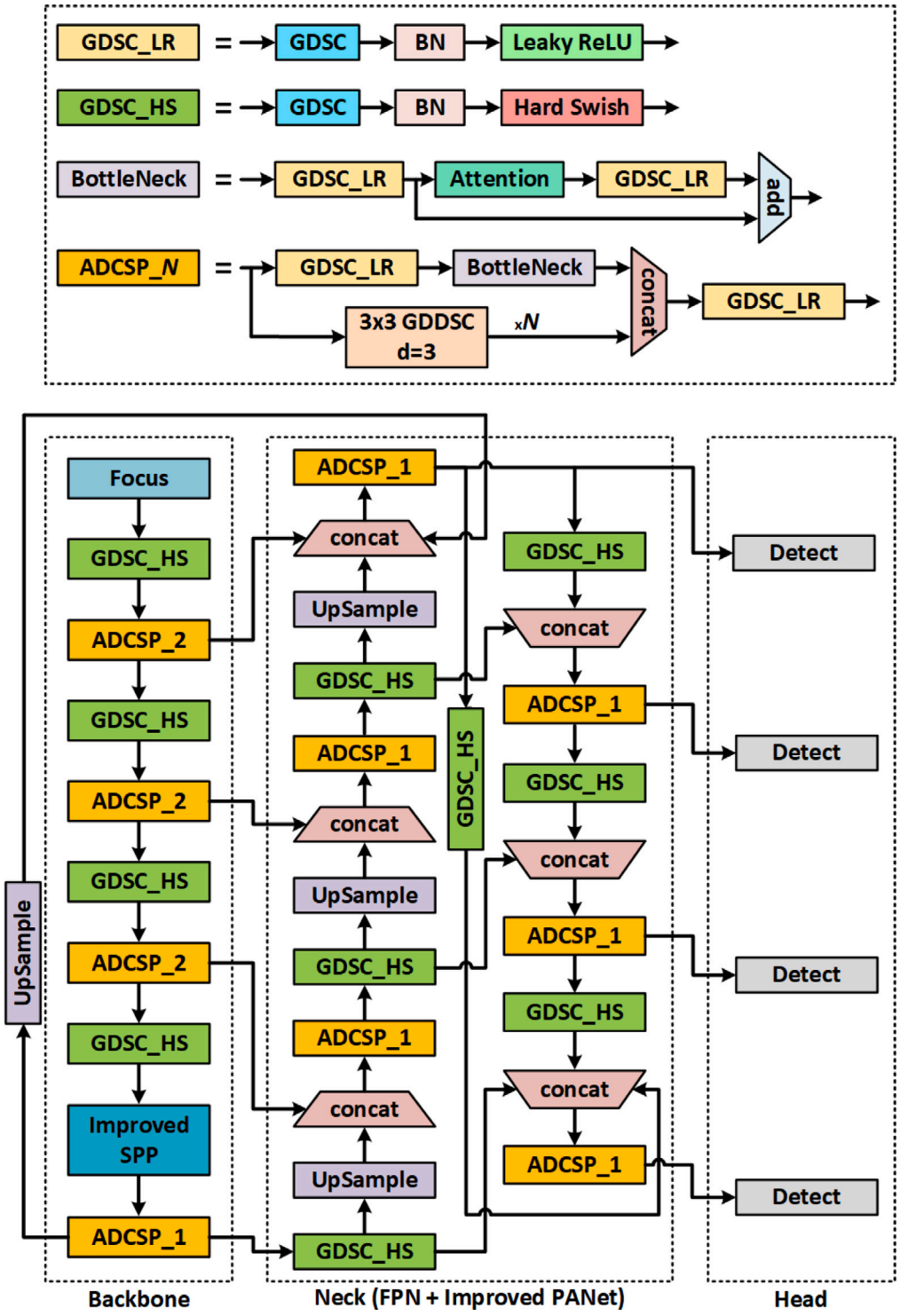

Fig. 5. Detailed structure of the proposed IS-YOLOv5 model.

problem, mathematically defined as

$$\min_{\mathcal{W}'} \quad |\mathcal{L}(\mathcal{W}') - \mathcal{L}(\mathcal{W})|, \\ \text{s.t.} \quad \#(\mathcal{W}') = \#(\mathcal{W}) \times (1-p), \tag{12}$$

where $\#(\mathcal{W}')$ and $\#(\mathcal{W})$ indicate the number of parameters in $\mathcal{W}'$ and $\mathcal{W}$, respectively. In the optimization problem, the evaluation of the loss function $\mathcal{L}(.)$ is difficult to solve directly for networks of any reasonable size (Anwar & Sung, 2016). Therefore, starting with a full parameter set $\mathcal{W}$, we identity and versatilely prune less-essential convolution kernels at each iteration, which eventually produces a parameter set $\mathcal{W}'$.

During the pruning implementation, the change in the loss function $\mathcal{L}(.)$ can be approximated using the Taylor expansion to set the criterion when pruning specific convolution kernels. In particular, we utilize $\mathcal{G} = \{g_1, \ldots, g_k, \ldots, g_K\}$ to describe the set of feature maps, where g_k denotes the feature map generated from convolution kernel f_k. As the loss function $\mathcal{L}(.)$ is dependent equally on the parameter set $\mathcal{W}$ and the feature map set $\mathcal{G}$, we can infer that $\mathcal{L}(\mathcal{W}) = \mathcal{L}(\mathcal{G})$. Therefore, the change

in the loss function $\Delta\mathcal{L}_k$ after pruning the kernel f_k can be defined as

$$\Delta\mathcal{L}_k = |\mathcal{L}(\mathcal{G}|g_k = 0) - \mathcal{L}(\mathcal{G}|g_k = \epsilon)|. \tag{13}$$

where $\mathcal{L}(\mathcal{G}|g_k = 0)$ and $\mathcal{L}(\mathcal{G}|g_k = \epsilon)$ denote the loss value if the convolution kernel f_k is pruned and the loss value if it is not pruned, respectively. To compute the loss value $\mathcal{L}(\mathcal{G}|g_k = \epsilon)$ for the function $\mathcal{L}(\mathcal{G}|g_k)$ with respect to g_k by inputting images, we can define the first-order Taylor expansion at $g_k = \epsilon$ as

$$\mathcal{L}(\mathcal{G}|g_k) = |\mathcal{L}(\mathcal{G}|g_k = \epsilon) + \frac{\partial\mathcal{L}}{\partial g_k}\Big|_{g_k=\epsilon}(g_k - \epsilon) + \mathcal{M}_1(g_k)|, \tag{14}$$

where $\mathcal{M}_1(g_k)$ denotes the remainder of the first-order Taylor expansion. The remainder term $\mathcal{M}_1(g_k = 0)$ can be calculated using the Lagrange method as

$$\mathcal{M}_1(g_k = 0) = \frac{\partial^2\mathcal{L}}{\partial(g_k^2 = \zeta)} \cdot \frac{g_k^2}{2}. \tag{15}$$

where ζ is a real value between 0 and g_k. Assuming that a Taylor polynomial of first-degree is present in $\mathcal{L}(\mathcal{G}|g_k)$ near $g_k = 0$, we can approximate $\mathcal{L}(\mathcal{G}|g_k = 0)$ using the known $\mathcal{L}(\mathcal{G}|g_k = \epsilon)$. However, we ignore the first-degree remainder term $\mathcal{M}_1(g_k)$ because it requires a large number of calculations but also because the proposed IS-YOLOv5 model employs the Hard Swish and ReLU based activations, which will promote a smaller second-degree term when ζ is close to 0. Hence, we choose to neglect this remainder term, the approximation can be written as

$$\mathcal{L}(\mathcal{G}|g_k = 0) \approx \mathcal{L}(\mathcal{G}|g_k = \epsilon) - (\frac{\partial\mathcal{L}}{\partial g_k}g_k)\Big|_{g_k=\epsilon}, \tag{16}$$

Substituting Eq. (16) into Eq. (13), we can rewrite Eq. (13) as

$$\Delta\mathcal{L}_k = |(\frac{\partial\mathcal{L}}{\partial g_k}g_k)\Big|_{g_k=\epsilon}|. \tag{17}$$

As observed, our scheme only requires a gradient of the loss function $\mathcal{L}(.)$ with respect to g_k and a feature map g_k to approximate the change in the loss function $\Delta\mathcal{L}_k$ after pruning the convolution kernel f_k. Notably, the backpropagation algorithm can easily compute $\Delta\mathcal{L}_k$ from the same computations. Moreover, the contribution C_k of kernel f_k is correlated positively to $\Delta\mathcal{L}_k$, which is calculated using the gradient and the feature map, defined as

$$C_k = \frac{1}{Q}\sum_q |\frac{\partial\mathcal{L}}{\partial g_{k,q}}g_{k,q}|. \tag{18}$$

where Q represents the number of feature map elements and $g_{k,q}$ denotes the qth element of the kth feature map matrix. In the presence of a batch of multiple images, the ranking criterion is measured separately for each image and averaged across all images. Intuitively, if $|\frac{\partial\mathcal{L}}{\partial g_k}g_k|$ is small and close to 0, the convolution kernel g_k must be sparse and makes a trivial contribution C_k, so it can be removed from the network. During the pruning process, we can effectively reduce this sparsity of convolution kernels in different layers, making the network small and dense.

Algorithm 1 presents the proposed VNP strategy. To minimize the damage induced by pruning to the network $\mathbb{N}$, we use the kernel pruning scheme in an iterative manner. Initially, we fine-tune the network $\mathbb{N}$ or its pruned version until it converges over the dataset D. Next, the contribution C_k is computed for each convolution kernel g_k, and the less-essential kernels that have low contribution are pruned. Then, the pruning rate p' is updated based on the number of pruned convolution kernels. Lastly, we repeat the above process until we reach a specified pruning rate p, at which point the pruned lightweight network $\mathbb{C}$ is obtained. Through this approach, the computational complexity and storage cost of the proposed network model are greatly reduced.

Algorithm 1 Versatile Network Pruning (VNP) Algorithm

Input:
Original network: $\mathbb{N}$
Training dataset: D
Number of neurons pruned per iteration: $\#(\mathcal{R})$
Predefined pruning rate: p
Kernel pruning rate: p'
Output:
Pruned network: $\mathbb{C}$

```
Begin
Initialize p' = 0
while p' < p do
    Perform fine-tuning on N or its pruned version over D
    Forward Propagation:
        Obtain feature map set G
    Backpropagation:
        Obtain |∂L/∂g_k| for each feature map g_k
    Compute C_k of each convolution kernel f_k (Eq. (18))
    Prune #(R) less-important convolution kernels based on C_k
    Update p'
end while
Return C
End
```

4. Experimental results

In this section, we describe the driving datasets, data augmentation techniques, training environment, and performance evaluation indicators. Thereafter, we verify the superiority of the proposed work through several experiments.

4.1. Dataset description

For most experiments, we choose BDD100K (Yu et al., 2020) as our primary dataset to benchmark the performance of the proposed framework. Besides, we analyze the detection performance of traffic signs using STSD (Larsson & Felsberg, 2011) and GTSDB (Stallkamp, Schlipsing, Salmen, & Igel, 2011) datasets. At the same time, we use BSTLD (Behrendt, Novak, & Botros, 2017) and S2TLD (Yang et al., 2022) datasets for evaluating the detection performance of traffic lights. It should be noted that STSD, GTSDB, BSTLD, and S2TLD are challenging datasets for road traffic target detection.

(1) Swedish Traffic Signs (STSD) dataset contains more than 20 K images from diverse highway and city environments, with 3488 annotated traffic signs.

(2) German Traffic Sign Detection Benchmark (GTSDB) dataset consists of around 10 K images from different road scenarios like rural, urban, and highway, with 1206 annotated traffic signs.

(3) Bosch Small Traffic Lights (BSTLD) dataset contains more than 13 K images from diverse driving scenes, including busy street, suburban, and inner-city at varying traffic density, with 24,242 annotated traffic lights.

(4) SJTU Small Traffic Light (S2TLD) dataset consists of around 6 K images from various lighting, weather, and traffic conditions, such as dense traffic, crowded street, and inner-city, with 14,130 annotated traffic lights.

(5) Berkeley DeepDrive (BDD100K) dataset comprises 100 K self-driving images from different traffic environments in diverse weather scenarios like residential, streets, gas stations, tunnels, highways, and parking lots at various day and night times. This dataset is split into training set, testing set, and validation set as 70:20:10 with ten road object classes.

A network will benefit from the balanced distribution of classes when extracting global features from a large number of instances. In contrast, when the number of instances is small, the ability of the backbone network to extract features will be greatly reduced. To better understand this, Fig. 6 depicts the distribution of the detection bounding box of various objects in the BDD100K dataset. As shown, the BDD100K dataset contains a variety of different-sized objects, but the majority of the distribution is made up of small objects. Because the remaining four datasets contain small objects, the spread of their

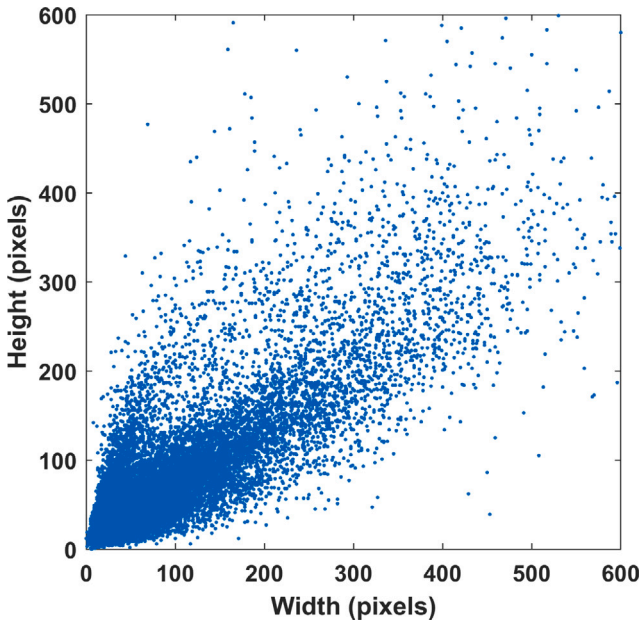

Fig. 6. Bounding box size distribution of BDD100K dataset.

distribution will be small. Therefore, for several categories, we consider objects large if their occupying area is more than 112 × 112 pixels, small if it is less than 48 × 48 pixels, and medium if it lies between the two thresholds. In this evaluation, we mainly focus on small objects like traffic lights and traffic signs. More than 80% of these objects have an area of less than 48 × 48 pixels.

4.2. Data augmentation

To enhance the quality of the training data, we apply a series of data augmentation techniques. Through this, a network model can learn the characteristics of objects in different scales, lightening, and angles, which can improve the model generalization performance on the unseen data. Among several data augmentation methods (Mahaur et al., 2022; Xiao et al., 2022), we adopt image displacement, linear scaling, horizontal flipping, motion blurring, uniform cropping, and noise adding. In addition, we use Mosaic data augmentation (Bochkovskiy et al., 2020), which allows us to train four images instead of one image. The benefit is that it enables training over a single GPU. Fig. 7 describes the working process of Mosaic data augmentation. To provide a fair comparison of the experimental results, we apply data augmentation techniques to all object detection models.

4.3. Training setup

The hardware environment comprises Intel i9-10900K CPU and NVIDIA Quadro RTX 5000 GPU. At the same time, the software environment is PyTorch v1.8.0 on Ubuntu 18.04.5 OS. We use the SGD optimizer for learning and updating the parameters through the training procedure (Jocher et al., 2022). A few hyperparameters are set as: the learning rate initialized to 0.01, the training batch size to 4, the momentum to 0.948, the weight decay to 0.0001, and the number of epochs to 100. We leave the remaining configurations as default to the original YOLOv5 model. After obtaining the trained network, we iteratively fine-tune the detection model with a learning rate of 0.001 and keep the remaining hyperparameters unchanged.

As the scale of objects in different datasets varies significantly, we use variable-sized anchors to minimize the problem of object scale imbalance. In YOLOv5, this approach applies the K-means clustering algorithm on a specific dataset to find the optimal anchor size. However, the traditional K-means algorithm is sensitive to initial clustering seeds and outliers (Gupta, Trivedi, & Prasad, 2022), which influences the detection performance of the model. Accordingly, in IS-YOLOv5, we use the K-means++ (Wang, Yeh, & Liao, 2021) clustering technique to overcome these limitations and ease the training process. Fig. 8 plots a schematic of the Intersection Over Union (IoU) calculated by K-means and K-means++ clustering algorithms on different training sets. The IoU values corresponding to the sets show an upward trend with the increase in K-values, implying that higher K-values increase the quality of the obtained anchor size for various objects. However, K-means++ exhibits a better clustering effect than K-means and is thus used as an anchor box clustering algorithm in the experiment. We perform anchor size clustering corresponding to four detection heads in our IS-YOLOv5, with each head assigned three scaling ratios, resulting in a total of twelve anchor sizes. Table 1 summarizes the anchor sizes for four detection heads obtained by the K-means++ algorithm on different training sets. Focusing on the multi-scale object detection problem, we use large anchors on the high-resolution feature map and small anchors on the low-resolution feature map.

Table 1
Results of anchor sizes using K-means++ algorithm corresponding to different datasets.

Dataset	Anchor sizes
STSD, GTSDB, BSTLD, S2TLD	(2, 5), (3, 8), (4, 6) (4, 15), (7, 10), (9, 7) (8, 28), (11, 15), (16, 23) (25, 37), (47, 61), (95, 135)
BDD100K	(3, 7), (5, 8), (7, 11) (13, 21), (9, 34), (24, 16) (36, 45), (42, 91), (73, 102) (115, 134), (156, 217), (209, 342)

4.4. Evaluation metrics

The mean Average Precision (mAP) is the most common evaluation index used to evaluate the detection performance of the network (Badue et al., 2021; Zou et al., 2019). mAP considers both Precision (P) and Recall (R), defined in Eqs. (19) and (20), respectively. AP is the average of precision at different recall values obtained by Precision–Recall (PR) curve under a given threshold, defined in Eq. (21). The IoU is the degree of overlap between the predicted box and the actual box, which is set to IoU ≥ 0.50. Finally, mAP is the mean of AP over all classes, defined in Eq. (22). In addition, we track computational complexity in terms of Floating Point Operations (FLOPs) and measure inference speed in terms of frames per second (FPS) for validating the model efficiency.

$$Precision\,(P) = \frac{TP}{TP + FP}, \tag{19}$$

$$Recall\,(R) = \frac{TP}{TP + FN}, \tag{20}$$

where TP denotes the number of detections in both the ground truth and the results. FP denotes the number of detections in the results only, not in the ground truth. FN denotes the number of detections in ground truth only, not in results.

$$Average\ Precision\,(AP) = \sum_{i} \left(R_{i+1} - R_i\right) \max_{R:\overline{R} \geq R_{i+1}} P(\overline{R}), \tag{21}$$

where $P(\overline{R})$ is the measured P at $\overline{R}$.

$$mean\ Average\ Precision\,(mAP) = \frac{1}{r} \sum_{j=1}^{r} AP_j. \tag{22}$$

where r is the number of classes and AP_j is the AP at class j.

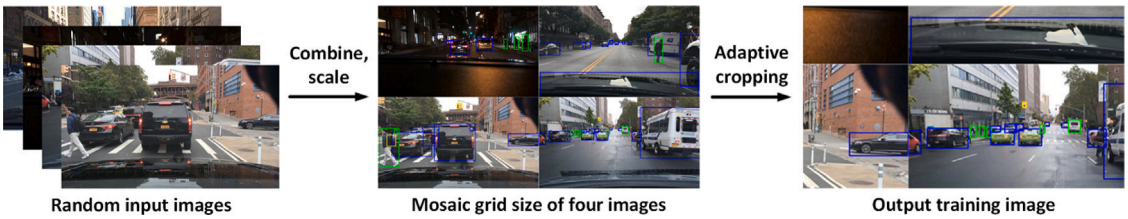

Fig. 7. Implementation of Mosaic data augmentation.

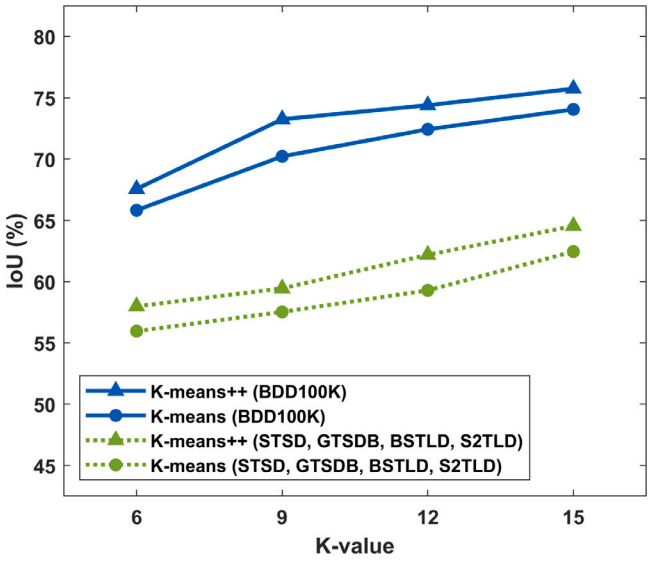

Fig. 8. Comparison of the two clustering algorithms on various datasets.

Table 2
Accuracy of traffic sign and traffic light of the proposed IS-YOLOv5 model on challenging datasets.

Model	$AP_{traffic\ sign}$		$AP_{traffic\ light}$	
	STSD	GTSDB	BSTLD	S2TLD
YOLOv4 (Bochkovskiy et al., 2020)	38.12	40.36	59.47	55.68
YOLOv5 (Jocher et al., 2022)	37.25	40.59	60.82	54.31
IS-YOLOv5	41.06	43.28	62.13	56.72

Table 3
Performance evaluation of the proposed IS-YOLOv5 model on BDD100K dataset.

Model	$AP_{traffic\ sign}$	$AP_{traffic\ light}$	mAP	FPS	GFLOPs
YOLOv4 (Bochkovskiy et al., 2020)	49.83	55.37	61.92	40.46	128.73
YOLOv5 (Jocher et al., 2022)	50.24	55.34	61.89	56.95	16.47
IS-YOLOv5	55.63	58.16	70.24	58.72	19.63

Table 4
Accuracy of the proposed IS-YOLOv5 model at three different scales on BDD100K dataset.

Model	mAP_{large}	mAP_{medium}	mAP_{small}
YOLOv4 (Bochkovskiy et al., 2020)	83.43	69.15	47.60
YOLOv5 (Jocher et al., 2022)	81.25	68.42	47.37
IS-YOLOv5	94.62	75.01	49.56

4.5. Performance comparison

In this set of experiments, we first demonstrate the efficacy of the IS-YOLOv5 model and the VNP scheme. We then validate the feasibility of the lightweight object detection framework created by combining the proposed techniques.

4.5.1. Effectiveness of the proposed IS-YOLOv5 architecture

4.5.1.1. Results on challenging datasets

We present empirical results for the detection of traffic lights and traffic signs in Table 2. As shown, our IS-YOLOv5 model improves the AP for traffic signs to 41.06% and 43.28% on STSD and GTSDB datasets, respectively. Meanwhile, on BSTLD and S2TLD datasets, the proposed model increases the AP for traffic lights to 62.13% and 56.72%, respectively. For a more comprehensive understanding, we show the proportion of correct, incorrect, and missed detections of the IS-YOLOv5 model for traffic lights and traffic signs in Fig. 9. As observed, most traffic objects are correctly detected even at lower widths, and the number of incorrect and missed detections decreases as the bounding box width increases. It is worth mentioning that our model is able to adapt to different datasets and outperform YOLOv4 and YOLOv5 models by a large margin, indicating better detection performance.

4.5.1.2. Results for small object detection

The detection results of the proposed IS-YOLOv5 model are shown in Table 3. Additionally, we plot the curves of change in mAP at different epochs for detection models in Fig. 10. It can be seen from the results that our IS-YOLOv5 increases the detection accuracy for small road objects like traffic signs and traffic lights to 55.63% and 58.16%, respectively. In all experiments, mAP is computed for all classes in the dataset. Compared with YOLOv5, the proposed model improves the mAP by 8.35% and inference speed by 1.77 FPS, with only 3.16 increase in computations. This verifies that architectural modifications can significantly improve the model performance at the cost of a few extra computations. Besides, compared with YOLOv4, our model increases the mAP by 8.32% and inference speed by 18.26 FPS at a much lower computational cost.

4.5.1.3. Results of different scale detection

We individually present the detection results for objects of different scales in Table 4. Moreover, the curves of mAP change at varying scales over different epochs are plotted in Fig. 11. As shown, the proposed IS-YOLOv5 model increases the mAP of large, medium, and small objects by 13.37%, 6.59%, and 2.19% compared to YOLOv5, respectively. Also, when compared to YOLOv4, our model increases the mAP of large, medium, and small objects by 11.19%, 5.86%, and 1.96%, respectively. Surprisingly, YOLOv4 has slightly higher mAP values than YOLOv5. It is worth emphasizing that the accuracy difference between large and small objects is quite significant. Through this set of experiments, we conclude that the detection accuracy of small objects can be improved without sacrificing the detection accuracy of medium and large objects.

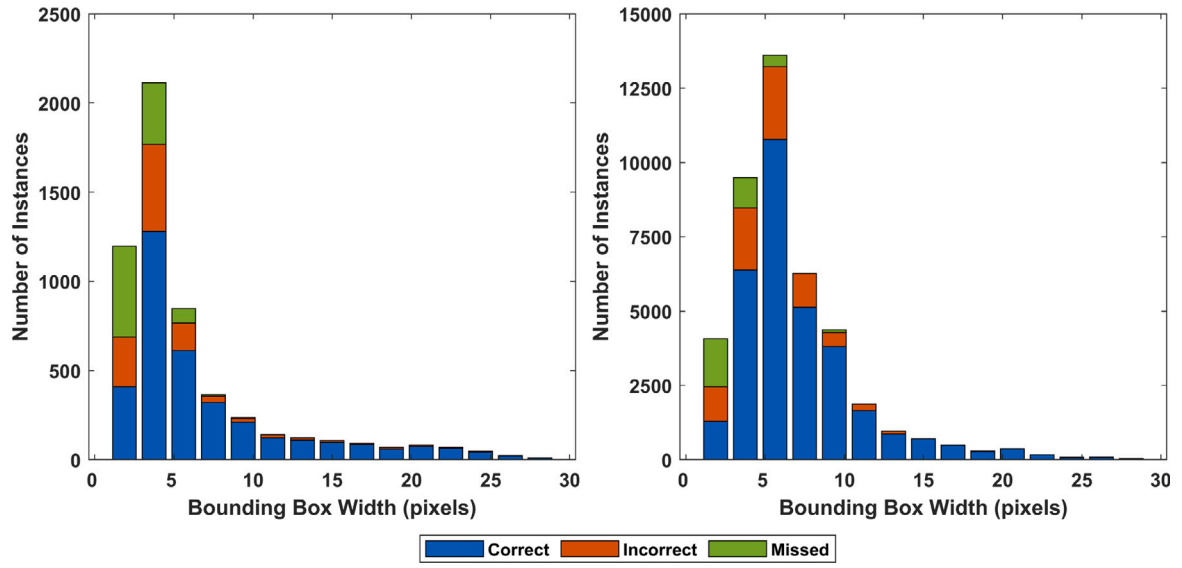

Fig. 9. Detection results of the proposed IS-YOLOv5 model (a) Traffic sign on STSD and GTSDB datasets (b) Traffic light on BSTLD and S2TLD datasets.

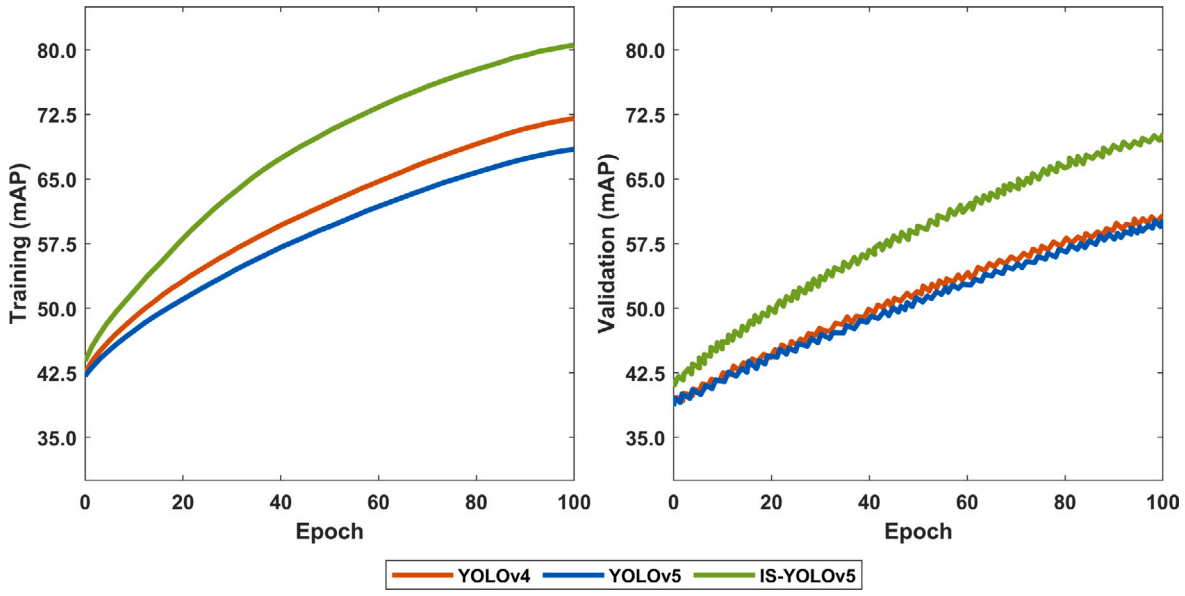

Fig. 10. Training and validation curves of the proposed IS-YOLOv5 model on BDD100K dataset.

4.5.1.4. Results of loss and PR curves

The validation loss curves for our IS-YOLOv5 model are visualized in Fig. 12. In particular, the bounding box, classification, and objectiveness loss curves are obtained during the training process. Additionally, Fig. 13 presents a comparison in terms of PR curves. It is generally believed that the closer the curve is to the upper right corner of the graph, the better the overall accuracy. In the case of small objects, the PR curves of IS-YOLOv5 have completely enclosed the PR curves of other object detection models. As observed, the curves of YOLOv4 and YOLOv5 mix and produce nearly the same value. The curves of Sparse R-CNN cover more area than the curves of Cascade R-CNN and Faster R-CNN. Meanwhile, the curves of Faster R-CNN and RetinaNet produce comparable results. The curves of SSD cover the smallest area for both objects, which indicates a high possibility of missed and false detection.

4.5.1.5. Results in diverse weather conditions

We test the applicability of the proposed IS-YOLOv5 model in dynamic weather scenarios like clear, overcast, partly cloudy, snowy, foggy, and rainy. Table 5 lists the generalization ability of the proposed model using AP values. The AP values of YOLOv5 are quite similar to those of YOLOv4 for simple weather scenarios, but as the weather becomes more complicated, YOLOv4 shows poor generalization for small objects. However, our IS-YOLOv5 surpasses both YOLOv4 and YOLOv5 in all road weather conditions. It is clear from the results that our model has a strong generalization performance even in very complex environments. This further indicates that the proposed modifications improve performance in complex weather scenarios, which can help extend the visual perception of autonomous driving systems.

To evaluate the effectiveness of the proposed IS-YOLOv5 model, we test our method in varying traffic environments. Figs. 14 and 15 compare the detection performance of small objects in varying traffic densities. As observed, most detection models miss traffic lights and traffic signs, even in low-traffic scenarios. For Faster R-CNN and RetinaNet, the performance of the true detected traffic objects is good, but the confidence is quite low. Furthermore, Sparse R-CNN obtains a fairly good detection accuracy than other R-CNN models. In contrast, the proposed modifications enhance the feature extraction capability

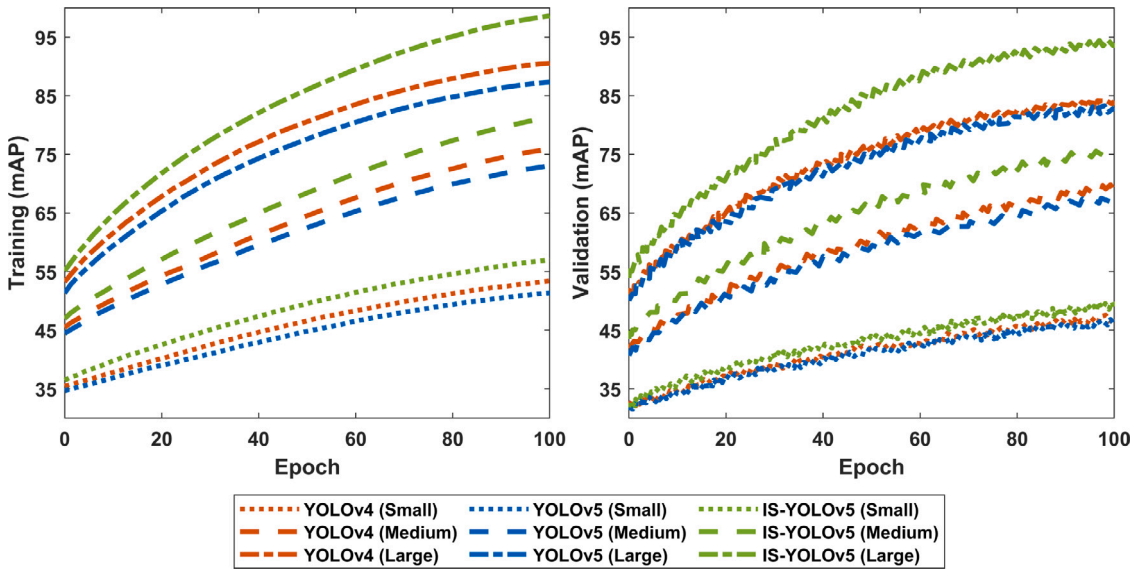

Fig. 11. Training and validation curves of the proposed IS-YOLOv5 model at varying scales on BDD100K dataset.

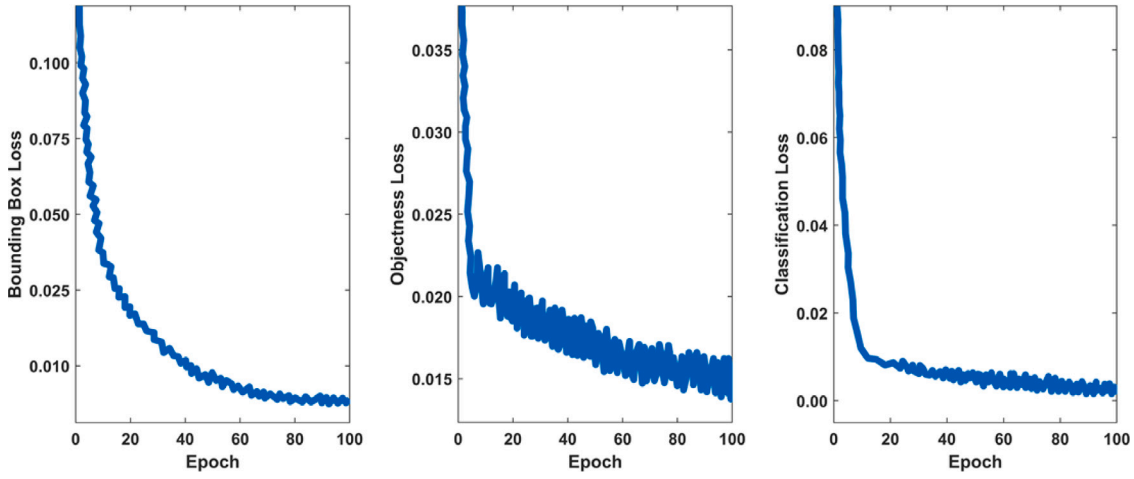

Fig. 12. Validation loss curves for the proposed IS-YOLOv5 model on BDD100K dataset.

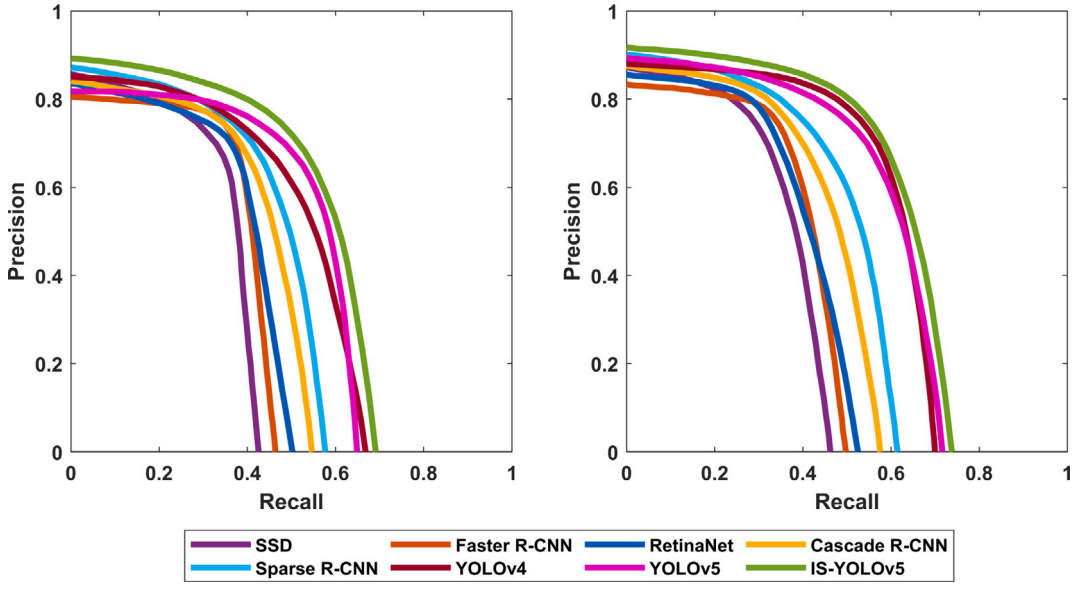

Fig. 13. Comparison of PR curves on BDD100K dataset (a) Traffic sign (b) Traffic light.

Table 5
Generalization ability of our IS-YOLOv5 model in different road weather conditions on BDD100K dataset.

Model		Weather scenarios					
		Clear	Partly Cloudy	Overcast	Foggy	Snowy	Rainy
YOLOv4 (Bochkovskiy et al., 2020)	$AP_{traffic\ sign}$	55.64	52.87	52.36	50.17	43.35	37.69
	$AP_{traffic\ light}$	58.79	57.23	56.85	52.39	51.08	47.53
YOLOv5 (Jocher et al., 2022)	$AP_{traffic\ sign}$	54.31	52.44	52.43	51.65	45.29	40.76
	$AP_{traffic\ light}$	57.94	57.51	56.47	53.84	52.63	49.61
IS-YOLOv5	$AP_{traffic\ sign}$	58.71	55.94	55.02	54.35	47.26	42.38
	$AP_{traffic\ light}$	62.80	61.73	59.58	56.97	54.32	51.67

of the IS-YOLOv5 model, thus exhibiting better confidence in detecting small traffic objects.

Likewise, in high-traffic scenarios, several representative models perform worse in detecting small objects. In particular, SSD misses the majority of road objects and has very low confidence in those that are detected. Moreover, YOLOv4 and YOLOv5 detect more small objects but have an inaccurate detection bounding box and a low confidence score for most traffic signs. Besides, our IS-YOLOv5 can accurately detect traffic lights and traffic signs with high confidence that other models fail to detect or incorrectly detect, demonstrating its superior performance in various traffic environments. With many small objects in a complex road scene at intersections, the majority of which are traffic signs and traffic lights. This type of detection scenario is an excellent test for assessing the detection performance of individual models.

4.5.1.6. Results with state-of-the-art detection models

In order to verify the superiority of the IS-YOLOv5 model architecture, we compare our method with several used object detectors. Table 6 represents the accuracy, speed, and complexity of different frameworks on the BDD100K dataset. The accuracy is reported for both default (IoU $\geq$ 0.50) and strict (IoU $\geq$ 0.75) metrics. To provide a fair comparison, all the methods are implemented under the same settings as described in their original articles. As shown, our IS-YOLOv5 achieves the best detection performances of 70.24% and 47.35% at a faster inference speed of 58.72 FPS compared to other approaches. Evidently, our model outperforms the benchmarks by a large margin at a relatively low computational cost. This proves that our proposed IS-YOLOv5 model achieves satisfactory performance results and is thus suitable for road object detection in autonomous driving applications.

4.5.2. Effectiveness of the proposed VNP scheme

4.5.2.1. Results by integrating with IS-YOLOv5 model

We apply our VNP technique to the trained IS-YOLOv5 detection model on the BDD100K dataset. Table 7 summarizes the performance of our method under different pruning rates. Note that the results in the first column are for the original unpruned model. As observed, the accuracy is reduced by only 1.27% when the pruning rate is set to 50%. However, it is worth noting that the accuracy loss is quite negligible at low pruning rates (<50%). While at high pruning rates (>50%), the detection speed, model size, and computational complexity have remarkably improved. More importantly, these comparative results validate our IS-YOLOv5+VNP method in terms of the trade-off between mAP and FPS.

4.5.2.2. Results for varying kernel sizes

To further assess the capabilities of the proposed method, we examine its performance when using various kernel sizes, i.e., 7 × 7, 5 × 5, and 3 × 3. Fig. 16 depicts the detection accuracy of our IS-YOLOv5 model under various kernel sizes at different pruning rates. As can be seen that at high pruning rates, the 3 × 3 kernel size yields more mAP than the 7 × 7 kernel size. Meanwhile, the mAP difference between the kernel sizes 5 × 5 and 3 × 3 is very small at low pruning rates. This shows the performance effect of our VNP scheme on varying kernel sizes in the IS-YOLOv5 model.

Table 6
Performance comparison with state-of-the-art detection models. The best and second best performances are denoted by **bold** and underlined fonts, respectively.

Model	mAP@0.50	mAP@0.75	FPS	GFLOPs
SSD (Liu et al., 2016)	52.40	28.71	54.93	95.86
YOLOR (Wang, Yeh, & Liao, 2021)	52.81	29.67	35.26	118.74
CenterNet (Duan et al., 2019)	53.76	30.59	42.15	63.08
YOLOF (Chen, Wang, et al., 2021)	54.29	31.72	38.47	94.65
Faster R-CNN (Girshick, 2015)	55.14	33.51	27.06	231.43
RetinaNet (Lin et al., 2017)	55.62	34.48	24.57	276.78
FCOS (Tian, Shen, Chen, & He, 2019)	56.34	35.79	28.63	201.45
Cascade R-CNN (Cai & Vasconcelos, 2018)	57.03	37.54	18.21	293.86
Sparse R-CNN (Sun et al., 2021)	58.44	36.82	21.36	170.75
YOLOv5 - S (Jocher et al., 2022)	61.89	39.53	56.95	**16.47**
YOLOv4 (Bochkovskiy et al., 2020)	61.92	39.57	40.46	128.73
Scaled YOLOv4 (Wang, Bochkovskiy, & Liao, 2021)	62.67	38.94	38.13	45.26
YOLOX (Ge et al., 2021)	63.19	40.56	52.47	27.83
YOLOv5 - M (Jocher et al., 2022)	64.72	41.23	51.48	50.94
YOLOv5 - L (Jocher et al., 2022)	65.15	42.75	37.22	114.87
EfficientDet (Tan, Pang, & Le, 2020)	67.09	44.19	28.53	26.64
YOLOv5 - X (Jocher et al., 2022)	67.83	45.62	14.69	218.75
IS-YOLOv5 (ours)	**70.24**	**47.35**	**58.72**	19.63

Fig. 14. Comparison of detection results of different models in a low traffic scene (a) SSD (Liu et al., 2016) (b) Faster R-CNN (Girshick, 2015) (c) RetinaNet (Lin et al., 2017) (d) Cascade R-CNN (Cai & Vasconcelos, 2018) (e) Sparse R-CNN (Sun et al., 2021) (f) YOLOv4 (Bochkovskiy et al., 2020) (g) YOLOv5 (Jocher et al., 2022) (h) IS-YOLOv5 (ours).

Table 7
Evaluation of our IS-YOLOv5 model when using the proposed VNP strategy at different pruning rates.

Pruning rate (%)	0	10	20	30	40	50	60	70	80	90
mAP	70.24	70.16	69.98	69.75	69.32	68.97	68.41	67.62	66.43	65.09
FPS	58.72	63.45	71.22	89.34	115.67	137.25	163.82	196.04	224.17	276.38
FLOPs (G)	19.63	18.27	16.05	13.28	10.43	8.61	5.74	3.57	1.85	0.94
Model size (MB)	43.28	40.74	35.86	31.07	26.59	23.44	20.63	14.16	9.03	5.17

4.5.2.3. Results with traditional pruning schemes

The performance of the proposed VNP scheme is compared to that of other traditional pruning methods. For this, we evaluate our Taylor criterion ranking-based kernel pruning technique with $L2$-norm and $L1$-norm based pruning approaches. The detection accuracy with various pruning techniques is illustrated in Fig. 17. Both $L2$-norm and $L1$-norm based approaches adopt the iterative pruning strategy. As observed, the proposed VNP scheme outperforms other approaches significantly. At low pruning rates, $L2$-norm based pruning achieves slightly better mAP than $L1$-norm based pruning. However, at high

Fig. 15. Comparison of detection results of different models in a high traffic scene (a) SSD (Liu et al., 2016) (b) Faster R-CNN (Girshick, 2015) (c) RetinaNet (Lin et al., 2017) (d) Cascade R-CNN (Cai & Vasconcelos, 2018) (e) Sparse R-CNN (Sun et al., 2021) (f) YOLOv4 (Bochkovskiy et al., 2020) (g) YOLOv5 (Jocher et al., 2022) (h) IS-YOLOv5 (ours).

pruning rates, both produce nearly the same mAP values. Besides, our VNP scheme achieves higher mAP than both pruning methods. In addition, when the pruning rate exceeds 30%, the decrease in mAP becomes more noticeable. This confirms that our VNP scheme is an effective technique for pruning redundant parameters in the network model.

4.5.2.4. Results by integrating with existing models

We integrate our VNP strategy into commonly used object detection models. Fig. 18 compares the mAP obtained by different models at various pruning rates. As shown, the difference between the mAP values is small for EfficientDet at different pruning rates. Furthermore, when the pruning rates are not high, YOLOv4 achieves the lowest loss difference in mAP compared to other models. At high pruning rates, Faster R-CNN and RetinaNet obtain similar mAP values. Meanwhile, SSD and Cascade R-CNN yield satisfying mAP at varying pruning rates. It is evident from the results that the proposed VNP scheme is capable of reducing the computational burden and can obtain good detection accuracy without significantly affecting the overall performance.

4.5.3. Effectiveness of the proposed lightweight framework

In Sections 4.5.1 and 4.5.2, we individually highlighted the effectiveness of IS-YOLOv5 model and VNP scheme, respectively. Both the proposed approaches are combined to create a lightweight framework called IS-YOLOv5+VNP. Finally, we present a comparison with the state-of-the-art lightweight object detection models in Table 8. All methods are fairly implemented using the default settings as per their original studies. It can be seen that our IS-YOLOv5+VNP approach has

Table 8
Performance comparison with state-of-the-art lightweight detection models. The best and second best performances are denoted by **bold** and underlined fonts, respectively.

Model	mAP@0.50	mAP@0.75	FPS	FLOPs (G)	Model Size (MB)
SqueezeDet (Wu et al., 2017)	40.13	21.64	57.86	12.35	20.79
Tinier-YOLO (Fang, Wang, & Ren, 2019)	48.51	26.07	65.23	6.38	25.64
SSDLite (Sandler et al., 2018)	52.93	29.84	70.47	**4.61**	21.38
YOLOv4-tiny (Jiang et al., 2020)	55.76	32.58	121.94	5.90	**16.87**
Pelee (Wang, Li, & Ling, 2018)	57.24	33.15	86.40	16.57	37.92
Tiny-DSOD (Li, Li, Lin, & Li, 2018)	58.72	33.76	51.94	18.26	32.50
ThunderNet (Qin et al., 2019)	60.19	38.42	79.15	7.93	24.81
RefineDetLite (Chen, Liu, Meng, Xiao, & Ju, 2020)	64.81	42.03	75.62	9.48	23.65
IS-YOLOv5+VNP ($p' = 30$)	**69.75**	**46.12**	89.34	13.28	31.07
IS-YOLOv5+VNP ($p' = 60$)	68.41	45.97	**163.82**	5.74	20.63

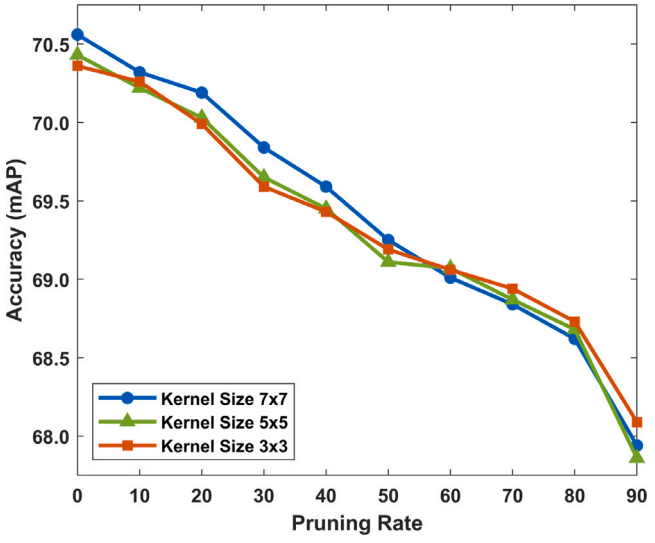

Fig. 16. Detection accuracy under various kernel sizes at different pruning rates.

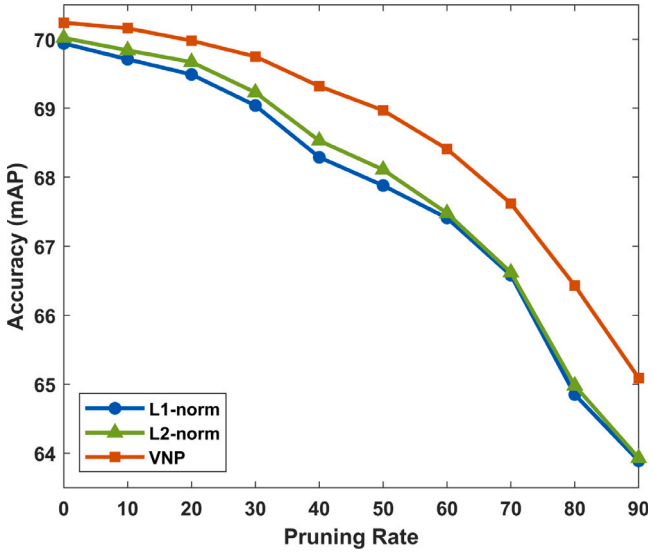

Fig. 17. Detection accuracy with traditional pruning approaches at different pruning rates.

higher mAP and FPS than other lightweight models, even after pruning a large number of network parameters. Besides, SSDLite and YOLOv4-tiny have low computational complexity and small model size, but their detection accuracy is very low. However, the proposed framework is able to preserve the detection performance at reasonable complexity and model size. This demonstrates that our proposed IS-YOLOv5+VNP method could meet the real-time requirements for object detection on the low-computing platform in autonomous driving systems.

5. Conclusion

In this article, we study and analyze the effect of different architectural modifications applied to the popular YOLOv5 structure for improving the detection performance of small-scale objects without compromising the detection accuracy of large and medium objects at low cost. To achieve this, we make structural refinements to optimize the flow of information through different network layers. Accordingly, we propose the IS-YOLOv5 model, which is capable of boosting the overall accuracy and detection speed without greatly increasing the network complexity. Furthermore, we reduce the computational burden of our IS-YOLOv5 model by proposing and applying the network pruning technique called the VNP scheme. By doing so, we minimize the size and complexity of the model while keeping the detection accuracy almost unchanged at fast inference speed. Through extensive experimentation, we individually and collectively validate the superiority of our approaches by highlighting the specific needs and limitations of autonomous driving scenarios. We also test the generalization applicability of the proposed method in complex road weather conditions. To the best of our knowledge, this is the first work to redesign individual modules in YOLOv5 while keeping the model complexity in mind in order to make it suitable for low-computing systems. As a result, we combine the IS-YOLOv5 model and the VNP scheme to develop an improved lightweight object detection framework. Using the IS-YOLOv5+VNP framework, we can accelerate model deployment and achieve real-time performance on the vehicle-mounted computing platform with resource constraints. Based on these insights, the current driving systems can be updated to detect small targets like traffic signs

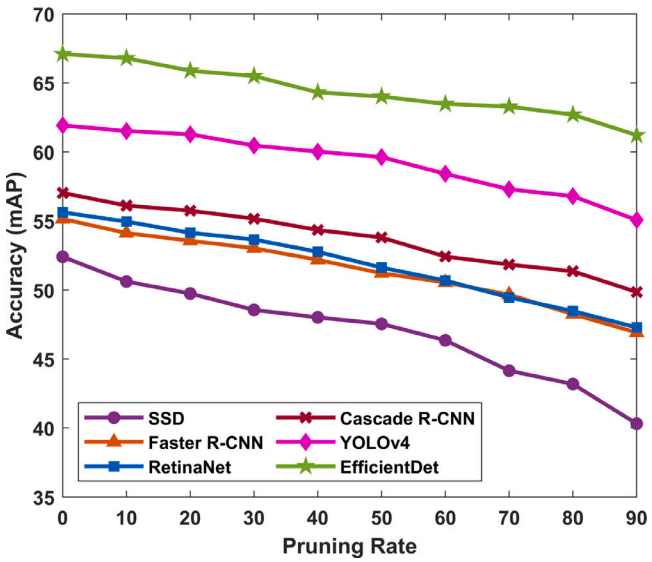

Fig. 18. Detection accuracy of existing detection models at different pruning rates.

and traffic lights in situations where existing models are incapable of detecting anything at all. Through our findings, the detection and perception robustness of an autonomous vehicle can be further extended, resulting in effective planning and decision-making.

CRediT authorship contribution statement

Bharat Mahaur: Conceptualization, Methodology, Data curation, Formal analysis, Investigation, Software, Visualization, Writing – original draft, Writing – review & editing. **K.K. Mishra:** Supervision, Resources, Validation, Writing – review & editing. **Anoj Kumar:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article

References

Anwar, S., & Sung, W. (2016). Compact deep convolutional neural networks with coarse pruning. arXiv preprint arXiv:1610.09639.

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., et al. (2021). Self-driving cars: A survey. *Expert Systems with Applications*, *165*, Article 113816.

Behrendt, K., Novak, L., & Botros, R. (2017). A deep learning approach to traffic lights: Detection, tracking, and classification. In *2017 IEEE international conference on robotics and automation* (pp. 1370–1377). IEEE.

Benjumea, A., Teeti, I., Cuzzolin, F., & Bradley, A. (2021). YOLO-z: Improving small object detection in YOLOv5 for autonomous vehicles. arXiv preprint arXiv:2112.11798.

Bie, M., Liu, Y., Li, G., Hong, J., & Li, J. (2022). Real-time vehicle detection algorithm based on a lightweight You-Only-Look-Once (YOLOv5n-L) approach. *Expert Systems with Applications*, Article 119108.

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

Cai, Y., Luan, T., Gao, H., Wang, H., Chen, L., Li, Y., et al. (2021). YOLOv4-5D: An effective and efficient object detector for autonomous driving. *IEEE Transactions on Instrumentation and Measurement*, *70*, 1–13.

Cai, Z., & Vasconcelos, N. (2018). Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6154–6162).

Chen, J., Jia, K., Chen, W., Lv, Z., & Zhang, R. (2022). A real-time and high-precision method for small traffic-signs recognition. *Neural Computing and Applications*, *34*(3), 2233–2245.

Chen, L., Lin, S., Lu, X., Cao, D., Wu, H., Guo, C., et al. (2021). Deep neural network based vehicle and pedestrian detection for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, *22*(6), 3234–3246.

Chen, C., Liu, M., Meng, X., Xiao, W., & Ju, Q. (2020). Refinedetlite: A lightweight one-stage object detection framework for cpu-only devices. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops* (pp. 700–701).

Chen, G., Wang, H., Chen, K., Li, Z., Song, Z., Liu, Y., et al. (2020). A survey of the four pillars for small object detection: Multiscale representation, contextual information, super-resolution, and region proposal. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Chen, Q., Wang, Y., Yang, T., Zhang, X., Cheng, J., & Sun, J. (2021). You only look one-level feature. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13039–13048).

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* (pp. 6569–6578).

Fang, W., Wang, L., & Ren, P. (2019). Tinier-YOLO: A real-time object detection method for constrained environments. *IEEE Access*, *8*, 1935–1944.

Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). Yolox: Exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440–1448).

Gupta, A., & Mahaur, B. (2021). An improved DV-maxHop localization algorithm for wireless sensor networks. *Wireless Personal Communications*, *117*, 2341–2357.

Gupta, A., Trivedi, A., & Prasad, B. (2022). Deployment and trajectory design of fixed-wing UAVs in NOMA assisted wireless networks. *Physical Communication*, *54*, Article 101789.

He, Y., Kang, G., Dong, X., Fu, Y., & Yang, Y. (2018). Soft filter pruning for accelerating deep convolutional neural networks. arXiv preprint arXiv:1808.06866.

Huang, Y., & Wang, Z. (2020). Multi-granularity pruning for deep residual networks. *Journal of Intelligent & Fuzzy Systems*, *39*(5), 7403–7410.

Jiang, Z., Zhao, L., Li, S., & Jia, Y. (2020). Real-time object detection method based on improved YOLOv4-tiny. arXiv preprint arXiv:2011.04244.

Jocher, G., et al. (2022). *ultralytics/yolov5: v6.1 - TensorRT, TensorFlow edge TPU and OpenVINO export and inference*. Zenodo, http://dx.doi.org/10.5281/zenodo.6222936.

Katsamenis, I., Karolou, E. E., Davradou, A., Protopapadakis, E., Doulamis, A., Doulamis, N., et al. (2022). TraCon: A novel dataset for real-time traffic cones detection using deep learning. arXiv preprint arXiv:2205.11830.

Khosravian, A., Amirkhani, A., Kashiani, H., & Masih-Tehrani, M. (2021). Generalizing state-of-the-art object detectors for autonomous vehicles in unseen environments. *Expert Systems with Applications*, *183*, Article 115417.

Larsson, F., & Felsberg, M. (2011). Using Fourier descriptors and spatial models for traffic sign recognition. In *Scandinavian conference on image analysis* (pp. 238–249). Springer.

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2016). Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710.

Li, J., Li, H., Chen, Y., Ding, Z., Li, N., Ma, M., et al. (2021). ABCP: Automatic block-wise and channel-wise network pruning via joint search. arXiv preprint arXiv:2110.03858.

Li, Y., Li, J., Lin, W., & Li, J. (2018). Tiny-DSOD: Lightweight object detection for resource-restricted usages. arXiv preprint arXiv:1807.11013.

Li, N., Song, F., Zhang, Y., Liang, P., & Cheng, E. (2022). Traffic context aware data augmentation for rare object detection in autonomous driving. arXiv preprint arXiv:2205.00376.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980–2988).

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., et al. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37). Springer.

Liu, K., Liu, W., Ma, H., Tan, M., & Gan, C. (2020). A real-time action representation with temporal encoding and deep compression. *IEEE Transactions on Circuits and Systems for Video Technology*, *31*(2), 647–660.

Liu, Y., Sun, P., Wergeles, N., & Shang, Y. (2021). A survey and performance evaluation of deep learning methods for small object detection. *Expert Systems with Applications*, *172*, Article 114602.

Mahaur, B., & Mishra, K. (2023). Small-object detection based on YOLOv5 in autonomous driving systems. *Pattern Recognition Letters*, *168*, 115–122.

Mahaur, B., Mishra, K., & Singh, N. (2023). Improved residual network based on norm-preservation for visual recognition. *Neural Networks*, *157*, 305–322.

Mahaur, B., Singh, N., & Mishra, K. (2022). Road object detection: a comparative study of deep learning-based algorithms. *Multimedia Tools and Applications*, *81*(10), 14247–14282.

Mishra, R., Gupta, H. P., & Dutta, T. (2020). A survey on deep neural network compression: Challenges, overview, and solutions. arXiv preprint arXiv:2010.03954.

Molchanov, P., Tyree, S., Karras, T., Aila, T., & Kautz, J. (2016). Pruning convolutional neural networks for resource efficient inference. arXiv preprint arXiv:1611.06440.

Niu, H., Hu, X., & Li, H. (2021). Improved YOLOv5 network-based object detection for anti-intrusion of gantry crane. In *2021 2nd international conference on control, robotics and intelligent system* (pp. 147–152).

Qin, Z., Li, Z., Zhang, Z., Bao, Y., Yu, G., Peng, Y., et al. (2019). ThunderNet: Towards real-time generic object detection on mobile devices. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 6718–6727).

Qin, Z., Zhang, P., Wu, F., & Li, X. (2021). Fcanet: Frequency channel attention networks. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 783–792).

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* (pp. 4510–4520).

Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2011). The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks* (pp. 1453–1460). IEEE.

Sun, P., Zhang, R., Jiang, Y., Kong, T., Xu, C., Zhan, W., et al. (2021). Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 14454–14463).

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* (pp. 10781–10790).

Tian, Z., Shen, C., Chen, H., & He, T. (2019). Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9627–9636).

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2021). Scaled-yolov4: Scaling cross stage partial network. In *Proceedings of the IEEE/Cvf conference on computer vision and pattern recognition* (pp. 13029–13038).

Wang, R. J., Li, X., & Ling, C. X. (2018). Pelee: A real-time object detection system on mobile devices. *Advances in Neural Information Processing Systems*, *31*.

Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2021). You only learn one representation: Unified network for multiple tasks. arXiv preprint arXiv:2105.04206.

Wu, B., Iandola, F., Jin, P. H., & Keutzer, K. (2017). Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 129–137).

Xiao, J., Guo, H., Zhou, J., Zhao, T., Yu, Q., & Chen, Y. (2022). Tiny object detection with context enhancement and feature purification. *Expert Systems with Applications*, Article 118665.

Yang, X., Yan, J., Liao, W., Yang, X., Tang, J., & He, T. (2022). Scrdet++: Detecting small, cluttered and rotated objects via instance-level feature denoising and rotation loss smoothing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., et al. (2020). Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2636–2645).

Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6848–6856).

Zhu, Y., & Yan, W. Q. (2022). Traffic sign recognition based on deep learning. *Multimedia Tools and Applications*, *81*(13), 17779–17791.

Zou, Z., Shi, Z., Guo, Y., & Ye, J. (2019). Object detection in 20 years: A survey. arXiv preprint arXiv:1905.05055.