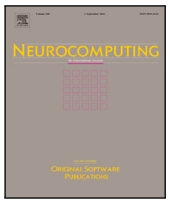
Neurocomputing 636 (2025) 129904

Contents lists available at ScienceDirect

Neurocomputing

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



YOLO-ELWNet: A lightweight object detection network ☆

Baoye Song [a], Jianyu Chen [a], Weibo Liu [b,*], Jingzhong Fang [b], Yani Xue [b], Xiaohui Liu [b]

[a] College of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao 266590, China
[b] Department of Computer Science, Brunel University London, Uxbridge, Middlesex, UB8 3PH, United Kingdom

ARTICLE INFO

Communicated by J. Hu

Keywords:
Object detection
YOLO
Lightweight network
Onboard device

ABSTRACT

This paper proposes a YOLO-based efficient lightweight network (YOLO-ELWNet) for onboard object detection based on the YOLOv3. A channel split and shuffle with coordinate attention module is developed in the backbone block, which effectively reduces the size of model parameters and computational cost while maintaining the detection accuracy. A new feature fusion network is proposed in the neck block, where a cross-stage partial with efficient bottleneck module is put forward to improve the feature extraction ability and reduce the computational cost. The Scylla intersection over union-based loss function is utilized in the head block, which accelerates the convergence speed of the YOLO-ELWNet. The effectiveness of the proposed YOLO-ELWNet is validated on the open source KITTI vision benchmark. The performance of YOLO-ELWNet is superior to some mainstream lightweight object detection models in terms of detection accuracy and computational cost, which demonstrates its applicability for resource-constrained onboard object detection.

1. Introduction

Object detection plays a critical role in a variety of fields such as autonomous driving, face recognition, and robot vision [1–4]. In the past few decades, a number of object detection techniques have been developed based on handcrafted features acquired through expert knowledge [5–7]. With the rapid development of deep learning techniques, modern object detection techniques have been introduced based on the convolutional neural network (CNN). Currently, popular CNN-based object detection methods can be classified into two categories: one-stage object detectors and two-stage object detectors [8].

One-stage object detection methods predict the class probability of the object from the candidate boxes directly. Some widely-used one-stage object detectors include the single shot multibox detector [8], the YOLO methods [9], and the RetinaNet [10], etc. The main procedure of existing two-stage object detectors can be summarized into two steps: (1) extracting candidate boxes from the image, and (2) obtaining detection results based on the candidate regions. Representative two-stage object detection methods include the regions with CNN features (R-CNN) [11], the spatial pyramid pooling network [12], the fast R-CNN [13], the faster R-CNN [14], and the mask R-CNN [15]. Two-stage object detection methods exhibit high recognition accuracy with the sacrifice of recognition efficiency. Substantial computational resources are required to build an effective two-stage object detector, which makes it unsuitable for onboard object detection on low-performance embedded devices. Compared with the two-stage methods, the one-stage object detection methods exhibit faster detection speed.

Due to the proper balance between the detection speed and accuracy, the YOLO methods have been successfully exploited in real-world object detection [16–18]. Unfortunately, limited by the computational resources and hardware storage, it is difficult to deploy the original YOLO methods to onboard devices. To tackle the resource-constrained object detection problem, designing lightweight networks has become an effective way to build an efficient onboard object detector with satisfactory detection accuracy while requiring relatively less computational resources [19,20].

Up to now, many YOLO-based lightweight networks have been developed for resource-constrained applications. For example, the YOLOv3-tiny algorithm has been proposed in [19] which includes fewer convolutional layers than the original YOLOv3, thereby requiring much less hardware memory. In [20], an enhanced YOLOv3-tiny algorithm has been presented to address the low detection accuracy problem in vehicle detection. In [21], an anchor configuration strategy has been presented in the YOLOv4-tiny to address the issue of omitting small objects for mobile device object detection. Recently, a

☆ This work was supported in part by the Natural Science Foundation of Shandong Province of China under Grant ZR2023MF067 and the National Natural Science Foundation of China under Grant 61933007.

* Corresponding author.
E-mail address: weibo.liu2@brunel.ac.uk (W. Liu).

https://doi.org/10.1016/j.neucom.2025.129904
Received 8 September 2024; Received in revised form 2 February 2025; Accepted 1 March 2025
Available online 19 March 2025

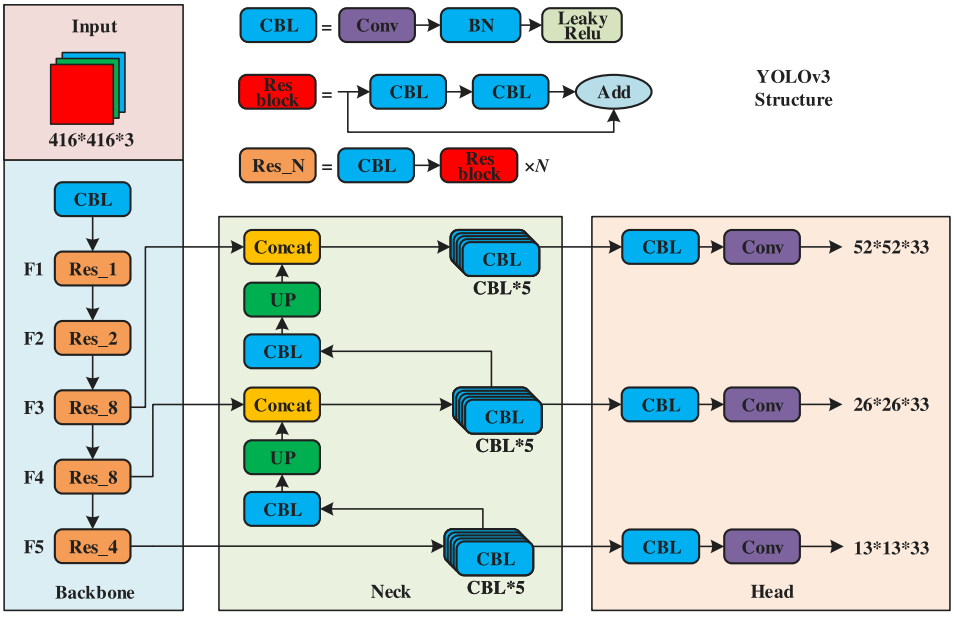


Fig. 1. Network structure of the YOLOv3.

modified YOLOv4-tiny has been introduced in [22] by integrating the self-attention mechanism and spatial pyramid pooling scheme, which achieves higher detection accuracy than the original YOLOv4-tiny while maintaining a consistent detection speed. More recently, an improved YOLOv5-based lightweight network has been developed in [23] for real-time detection of insulator failures on mobile devices. Very recently, a YOLOv5n-based lightweight network has been introduced in [24] for railway fastener inspection and localization.

Note that existing lightweight object detection methods still face the following challenges: (1) how to achieve a proper balance between the detection accuracy and the detection speed for onboard object detection? (2) how to maintain the generalization ability of the onboard object detector? and (3) how to build a reliable onboard object detector for handling complex scenes with limited resources? In light of the above-mentioned challenges, the YOLO-based efficient lightweight network (YOLO-ELWNet) is proposed in this paper based on the original YOLOv3 for onboard object detection. In the introduced YOLO-ELWNet, we modify the backbone, neck, and head blocks of the original YOLOv3 in order to boost the performance of the detector in terms of detection accuracy and detection speed under resource-constrained conditions.

The main contributions of this paper can be highlighted as follows.

(1) A channel split and shuffle with coordinate attention (CSS-CA) module is proposed for the backbone of the YOLO-ELWNet, thereby significantly reducing the parameter size and computational cost of the network while maintaining high detection accuracy.
(2) A feature pyramid network with cross stage partial (FPN-CSP) block is developed for the neck of YOLO-ELWNet, where a novel cross stage partial with efficient bottleneck (CSP-EB) module is designed to reduce the memory access cost (MAC) of the YOLO-ELWNet.
(3) The Scylla intersection over union (SIoU) loss function is employed to update the head block, which evidently accelerates the convergence speed of the YOLO-ELWNet.
(4) Experimental results demonstrate that the YOLO-ELWNet outperforms some currently popular lightweight networks and object detectors on the KITTI vision benchmark.

The remainder of this paper is organized as follows. Section 2 presents the preliminaries of the original YOLOv3. A detailed introduction of the proposed YOLO-ELWNet is presented in Section 3. Experimental results and discussions are reported in Section 4. Finally, the paper ends with the conclusion and future work in Section 5.

2. YOLOv3 preliminaries

YOLOv3 is a popular object detection method which is capable of detecting several objects with a single inference, thereby demonstrating fast detection speed [2,3,17]. The structure of the YOLOv3 is shown in Fig. 1, which includes four components, i.e., the input block, the backbone block, the neck block, and the head block. To be specific, the backbone block is the main feature extraction network. The neck block is the intermediate feature fusion network, and the head is the prediction block.

2.1. Input

In the original YOLOv3, the input size of the network is 416 × 416. To prevent distortion, gray bars are added to the images, which can effectively increase the detection accuracy and improve the computational speed of the detector. Usually, data augmentation techniques (such as random flipping, random scaling, and color space distortion) are exploited to expand the training set with the purpose of improving the generalization ability of the developed model.

2.2. Backbone

In YOLOv3, the DarkNet53 is employed as the backbone [17]. As shown in Fig. 1, the DarkNet53 begins with a combination of convolution-batch normalization-leaky ReLU (CBL) module to adjust the channel dimension to 32. That is to say, each CBL module consists of a convolutional layer, a batch normalization layer, and an activation function layer. Next, one CBL module (which is used to further downsample the feature map and double the channel dimension) is combined with a residual block (Resblock) to generate the feature map F1. The combination is repeated by connecting one CBL module with 2, 8, 8, and 4 Resblocks, respectively, resulting in feature maps F2, F3, F4, and F5. The obtained feature maps are then fed into the neck block.

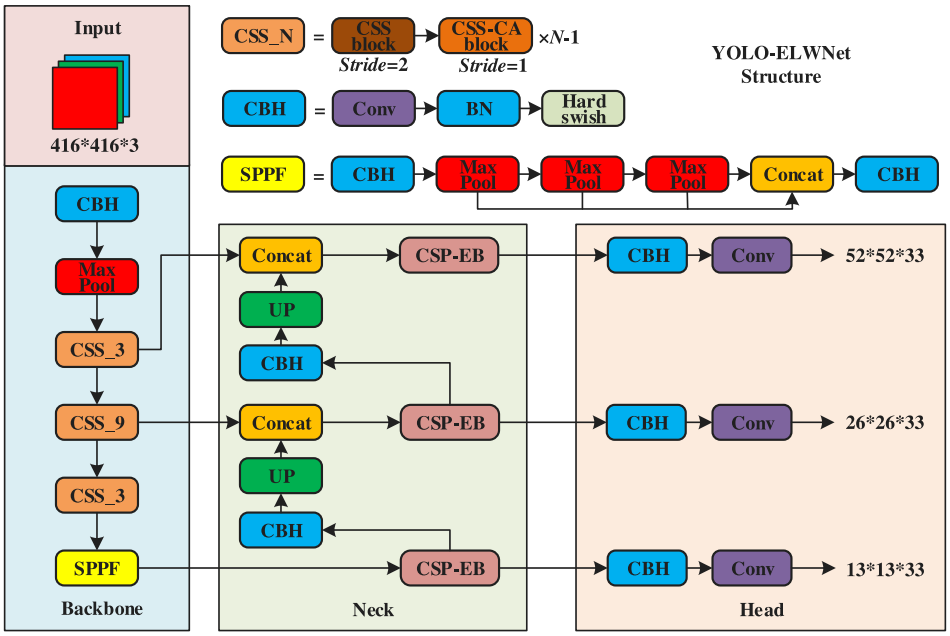


Fig. 2. Network structure of the YOLO-ELWNet.

2.3. Neck

The feature pyramid network (FPN) is employed as the neck block of YOLOv3, where the feature map F5 containing high-level semantic information goes through five CBL blocks. The resulting feature map is then split into two branches. One branch serves as the direct output, while the other branch undergoes a single CBL module and is upsampled before being concatenated with the F4 feature map. Similarly, both F3 and F4 in the YOLOv3 network have similar structures, which is depicted in Fig. 1. As such, the semantic information is propagated from lower-level feature maps to higher-level feature maps, facilitating feature fusion and enhancing the expressive power of the features.

2.4. Head

The prediction head utilizes anchor boxes to compute the center coordinates (e.g., width and height) of the predicted bounding box. The logistic function is employed to predict the object's class probability, and the intersection over union (IoU) is used to calculate the confidence score. Finally, a loss function (which incorporates target localization offset loss, target confidence loss, and target classification loss) is designed to measure the discrepancy between predicted boxes and ground truth boxes, thereby retaining the predicted boxes with the highest confidence scores and filtering out redundant objects.

Remark 1. The YOLOv3 is an efficient and practical method among the YOLO family due to the fast detection speed. The subsequent YOLO variants are usually optimized based on the network structure of the YOLOv3. The main limitation of the YOLOv3 lies in the "slightly" lower detection accuracy in comparison with other YOLO algorithms. Owing to its merits, we design a new lightweight network by using YOLOv3 as the baseline in this paper.

3. YOLO-ELWNet

In this paper, a YOLO-based efficient lightweight network (YOLO-ELWNet) is expressly designed for resource-constrained devices and mainly undergoes improvements on three aspects, including the backbone, neck, and head of the YOLO-ELWNet. The network structure of the proposed YOLO-ELWNet is depicted in Fig. 2.

3.1. Lightweight backbone

In the backbone block, a convolution-batch normalization-hardswish (CBH) module is employed to reduce the computational burden. The Hardswish activation function is utilized in the CBH module to improve the training efficiency [25]. A CSS-CA module based on the coordinate attention (CA) mechanism [26] is proposed to replace the Resblock in the original YOLOv3 backbone. The spatial pyramid pooling-fast (SPPF) module [27] is applied to enhance the detection speed of the backbone block.

3.1.1. CBH module

In the CBL module of the YOLOv3, the leaky ReLU is chosen as the activation function [17]. To tackle the computational limitations of embedded devices, the ReLU6 activation function is frequently employed in lightweight networks, which is suitable for low-precision calculations. However, the ReLU6 activation function would face the gradient vanishing problem when the input value of the activation function exceeds 6. Note that the Swish activation function combines the advantages of both ReLU and Sigmoid functions, which is widely exploited in variant YOLO methods [28]. Owing to the characteristics of smooth curve and boundedness, the Swish activation function could prevent overfitting. But the exponential operations of the Swish activation function would bring computational burden on resource-constrained devices. Thus, by integrating the ReLU6 and Swish functions, the Hardswish activation function is put forward in [25].

In this paper, the CBH module is established based on the Hardswish activation function to replace the CBL module in the original YOLOv3. The expression of Hardswish activation function can be formulated as follows

$$\text{Hardswish}(x) = \frac{x \cdot \text{ReLU6}(x+3)}{6}. \tag{1}$$

Remark 2. In comparison with the ReLU6 and Swish activation functions with hard restrictions and expensive computational cost, the Hardswish activation function can offer a smooth and bounded curve, enabling effective gradient propagation during training while reducing the computational burden significantly in lightweight networks.

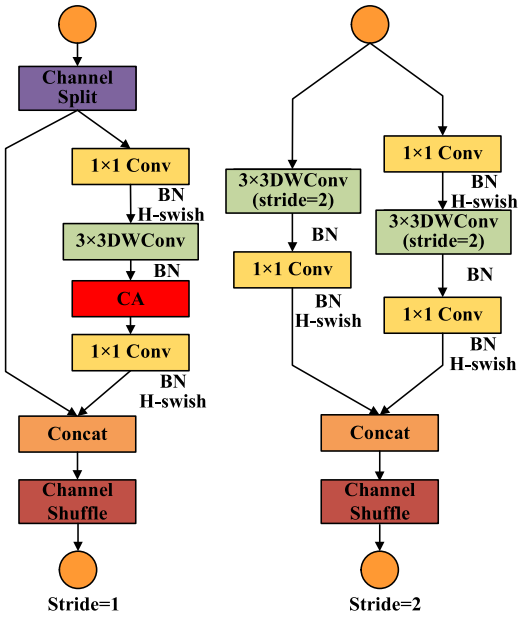


Fig. 3. Structure of the CSS-CA module.

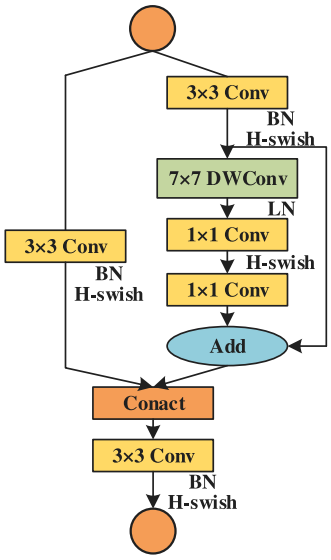


Fig. 4. Structure of the CSP-EB module.

3.1.2. CSS-CA module

In this paper, a CSS-CA module is deployed to replace the residual module in the original YOLOv3. Owing to its lightweight characteristic in ShuffleNetV2, the CSS module is employed for feature extraction [29, 30]. To further capture the spatial and channel feature, the CA module is embedded into the CSS module [26]. In fact, several attention modules have been successfully exploited in extracting spatial feature and channel feature. We choose the CA module due to its requirement of fewer parameters and less computational cost comparing with others. In this case, the introduced CSS-CA module could maintain the detection efficiency and requires less computational cost than the original ResNet block.

The structure of the CSS-CA module is shown in Fig. 5. To be specific, there are two different settings of the CSS-CA module depending on the stride size. When the stride is equal to 1, the CSS-CA module is split into two branches via channel split. As shown in Fig. 3, the input is fed into the convolution layer with 1×1 kernel, the depthwise convolution (DWC) with 3×3 kernel, the CA module, and the standard convolution layer with 1×1 kernel, which is then concatenated with the direct input of itself for channel shuffle. When the stride size is 2, the input is fed into two sets of convolution layers. The first set includes the DWC with 3×3 kernel and the standard convolution layer with 1×1 kernel. The second set consists of the convolution layer with 1×1 kernel, the DWC with 3×3 kernel, and the standard convolution layer with 1×1 kernel. Then, the outputs of the two sets are concatenated for channel shuffle. Note that the channel shuffle is employed in the CSS-CA module to ensure information transmission.

3.2. Improved neck

As displayed in Fig. 1, the FPN is used for the neck of YOLOv3. Unfortunately, the basic FPN suffers from weak feature extraction capabilities and high computational costs due to the simple stacking of CBL modules. To deal with the aforementioned problems, a CSP with efficient bottleneck (CSP-EB) module, inspired by CSPNet [31] and ConvNeXt [32], is proposed to replace five repeated CBL modules in FPN. The structure of the CSP-EB module is depicted in Fig. 4. After replacing the CBL modules in FPN with CSP-EB, a highly efficient feature fusion network called FPN-CSP is designed for the neck of YOLO-ELWNet.

As shown in Fig. 2, except for using the CSP-EB module to replace five CBL modules and change other CBLs to CBHs, FPN-CSP retains the same structure as the FPN in YOLOv3. The input is fed into two sets of convolution layers in the CSP-EB module depicted in Fig. 4. The first set includes the standard convolution layer with 3×3 kernel. The second set consists of a standard convolution layer with 3×3 kernel, a residual structure containing the DWC with 7×7 kernel, and two standard convolution layers with 1×1 kernel. Then, the outputs of the two sets are concatenated for channel adjustment using a standard convolution layer with 3×3 kernel. The CSP-EB structure effectively eliminates computational bottlenecks, allowing the network model to achieve higher performance with fewer computational resources. The concatenation technique deepens the connections between different layers and enhances the feature extraction capability of the CNN.

3.3. Modified head

The head of the proposed YOLO-ELWNet includes the CBH module followed by the convolution operator. Comparing with the baseline YOLOv3, the CBL modules are replaced by the CBH modules. The structure of the head is depicted in Fig. 2.

The SIoU loss function introduced in [33] is employed in the head of the YOLO-ELWNet. Different from the standard IoU loss function utilized in the YOLOv3, the SIoU loss function could alleviate the slow convergence and the wandering problem. The SIoU loss function is defined by

$$\mathrm{SIoU} = 1 - \mathrm{IoU} + \frac{\Delta + \Omega}{2}, \tag{2}$$

where Δ is the distance loss between the ground truth and predicted box; and Ω is the shape loss. The IoU loss is defined by

$$\mathrm{IoU} = \frac{\mathrm{B}^{\mathrm{gt}} \bigcap \mathrm{B}}{\mathrm{B}^{\mathrm{gt}} \bigcup \mathrm{B}}, \tag{3}$$

where B^{gt} and B indicate the areas of the ground truth and predicted boxes shown in Fig. 5. The distance loss Δ is given by

$$\Delta = \sum_{k=x,y} (1 - e^{-\gamma \rho_k}), \tag{4}$$

where $\rho_x = (\frac{B_{cx}^{gt} - B_{cx}}{C_w})$, $\rho_y = (\frac{B_{cy}^{gt} - B_{cy}}{C_h})$; $(B_{cx}^{gt}, B_{cy}^{gt})$ and (B_{cx}, B_{cy}) are the center coordinates of the ground truth and predicted boxes, respectively; C_w and C_h (which are shown in Fig. 5) are width and height of

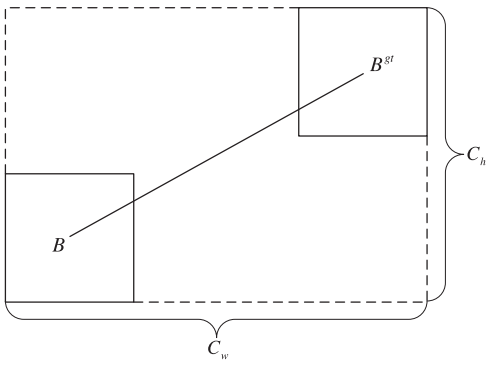


Fig. 5. Illustration of distance loss.

the minimum bounding rectangle for the ground truth and predicted boxes, respectively; γ is an intensity factor.

The shape loss Ω is presented by

$$\Omega = \sum_{k=w,h} (1 - e^{-\omega_k})^{\theta}, \tag{5}$$

where $\omega_w = \frac{|w^{gt} - w|}{\max(w^{gt}, w)}$, $\omega_h = \frac{|h^{gt} - h|}{\max(h^{gt}, h)}$; w^{gt} and h^{gt} are width and height of the ground truth box, respectively; w and h are width and height of the predicted box, respectively; θ is a hyperparameter representing the intensity of the shape loss.

4. Experiments

4.1. Dataset and experiment configuration

4.1.1. Dataset

In this paper, the KITTI dataset is employed to evaluate the performance of the developed YOLO-ELWNet [34]. KITTI is a public dataset consisting of real-world image data captured in various scenarios such as urban areas, rural areas, and highways. The KITTI dataset comprises a total of 7481 images with the size of 1242 × 375. The object types in the KITTI dataset include car, van, tram, truck, pedestrian, person-sitting, and cyclist. In this paper, we only focus on the pedestrians and vehicles categories. In this case, the objects of car, van, tram, and truck are merged into the car category, and the pedestrian, person-sitting, and cyclist images are merged into the pedestrian category. In the experiments, the images for training, validation, and testing are 6060, 673, and 748, respectively.

4.1.2. Experiment configuration

The configuration of the experimental environment of this paper is presented in Table 1. The CPU and GPU of the computer are configured as AMD Ryzen 9 5900X 12-Core Processor 3.70 GHz and NVIDIA GeForce RTX 3060 12G, respectively. The operating system is Windows 10 Professional Edition, and the integrated development environment is created by using Anaconda with Python 3.7.13 and PyTorch 1.7.1 deep learning framework. Additionally, the CUDA 11.0 computing platform and cuDNN 8.0.5 neural network library are used for acceleration in the experiments.

The Adam optimizer with a decay coefficient of 0.0005 is exploited for the model optimization. The input image is reshaped to 416 × 416, and the batch size is set as 32. The initial learning rate is set as 0.001. After each epoch, the learning rate decreases by a factor of 0.94 in a total of 100 epochs.

Table 1
Experimental environment.

Name	Configuration
CPU	AMD Ryzen 9 5900X 12-Core
GPU	NVIDIA GeForce RTX 3060 12G
Operating System	Windows 10 Professional Edition
Deep Learning Frame	Pytorch 1.7.1
Development Environment	Python 3.7.13, CUDA 11.0
GPU-accelerated Library	cuDNN 8.0.5

4.2. Evaluation metrics

In this paper, mean average precision (mAP), number of floating-point operations (FLOPs), number of parameters (Params), and frames per second (FPS) are used to evaluate the object detection methods [35]. mAP is calculated from the precision–recall curve based on precision and recall, representing the average precision across all classes. Due to limited computational resources of onboard devices, model complexity is another important metric that cannot be ignored except for detection accuracy. We use two commonly used metrics (e.g., FLOPs and Params) to measure the model complexity. Note that the FLOPs represent the theoretical computational load required by convolutions for training and running a model, which does not take the computational load required by other operations into account. Detection speed is an important metric which directly reflects whether the model meets the requirements for real-time detection or not. The FPS is employed as a direct metric to represent the detection speed of the model, while FLOPs serve as an indirect metric for the computational load. The positive and negative samples are determined using the IoU score. In detail, a sample is considered positive if the IoU score is larger than 0.5, and the sample is considered negative if the IoU score is below than 0.5. For fair comparisons, the performance of different models is tested on the same GPU (NVIDIA GeForce RTX 3060 12G) in the experiment.

4.3. Experimental results

4.3.1. Comparison of lightweight backbones

In this section, the backbone of YOLO-ELWNet (denoted by ELWNet-B) is compared with some other backbones, since the backbone plays a critical role in lightweight networks. In order to evaluate the performance of ELWNet-B, we replace DarkNet53 (which is the backbone of YOLOv3) by a few popular networks including MobileNetV3 [25], MobileNeXt [36], GhostNet [37], and EfficientNetV2 [38], and test all the detectors on the KITTI dataset. The network parameters for ELWNet-B are illustrated in Table 2.

The aforementioned four lightweight networks are used with their original structures, i.e., MobileNetV3-large for MobileNetV3, MobileNeXt-1.0 for MobileNeXt, GhostNet-1.0 for GhostNet, and the baseline for EfficientNetV2. Similar to DarkNet53, the obtained feature maps F3, F4, and F5 are fed into the neck as the input by using the four lightweight networks. For the backbone of YOLO-ELWNet, F3 corresponds to the 4th CSS-CA module, F4 corresponds to the 14th CSS-CA module, and F5 corresponds to the SPPF module.

The evaluation results of the utilized lightweight networks on the KITTI dataset are given in Table 3. It can be observed that ELWNet-B outperforms other lightweight networks in terms of the overall performance. Compared to the MobileNet series network, ELWNet-B demonstrates better detection accuracy with the sacrifice of a bit of detection speed. The mAP of ELWNet-B is 2.88% and 3.68% higher than that of the MobileNeXt and the MobileNetV3, respectively. Compared to GhostNet, ELWNet-B exhibits a 2.28% increase in mAP while a 7.22 decrease in FPS. Besides, EfficientNetV2 (which is the largest lightweight network) shows an 8.72 increase in FPS but a 2.54% lower mAP than ELWNet-B.

Table 2
Network parameters for ELWNet-B.

Input	Operator	Output channels	CA	Stride
416 × 416 × 3	Conv2d 6 × 6	64	–	2
208 × 208 × 64	Maxpool 3 × 3	64	–	2
104 × 104 × 64	CSS-CA	256	–	2
52 × 52 × 256	CSS-CA	256	✓	1
52 × 52 × 256	CSS-CA	256	✓	1
52 × 52 × 256	CSS-CA	256	✓	1
52 × 52 × 256	CSS-CA	512	–	2
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	512	✓	1
26 × 26 × 512	CSS-CA	1024	–	2
13 × 13 × 1024	CSS-CA	1024	✓	1
13 × 13 × 1024	CSS-CA	1024	✓	1
13 × 13 × 1024	CSS-CA	1024	✓	1
13 × 13 × 1024	SPPF	1024	–	–

Table 3
Comparison of lightweight backbones.

Backbone	mAP (%)	FPS	Params (M)	FLOPs (G)
ELWNet-B	81.12	49.00	27.67	22.05
MobileNetV3	77.44	72.98	23.18	17.42
MobileNeXt	78.24	62.09	22.73	18.28
GhostNet	78.84	56.22	22.25	16.80
EfficientNetV2	78.58	57.72	31.85	21.24

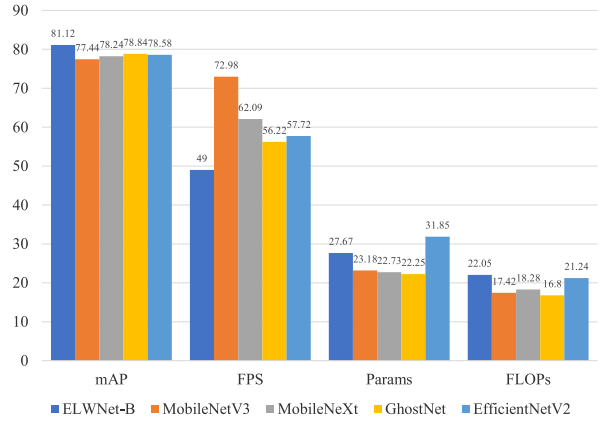


Fig. 6. Comparison of lightweight backbones.

For CNN-based object detectors, achieving a slight improvement in detection accuracy often comes at the cost of much slower detection speed and larger model size. The experimental results demonstrate that our ELWNet-B shows notable improvement in detection accuracy with slightly decreased detection speed and increased model size. To summarize, the proposed ELWNet-B exhibits the best overall performance among the five selected lightweight networks. For visualization, the bar chart is employed to compare the lightweight networks. The performance evaluation is displayed in Fig. 6, which offers an intuitive understanding of the performance comparison.

4.3.2. Ablation experiment

To evaluate the influence of the introduced modules, ablation experiments are conducted on the KITTI dataset for measuring the network performance. For fair comparison, all networks are trained and tested under the same hardware environment. The ablation experimental

Table 4
Results of ablation experiments.

Backbone	mAP (%)	FPS	Params (M)	FLOPs (G)
Baseline	79.17	45.00	61.53	65.60
Baseline+CSS	81.12	49.00	27.67	22.05
Baseline+CSS+SIoU	82.25	48.27	27.67	22.05
Baseline+CSS+SIoU+CSP	83.91	48.45	23.87	20.01
Baseline+CSS+SIoU+CSP+Aug	85.08	48.45	23.87	20.01

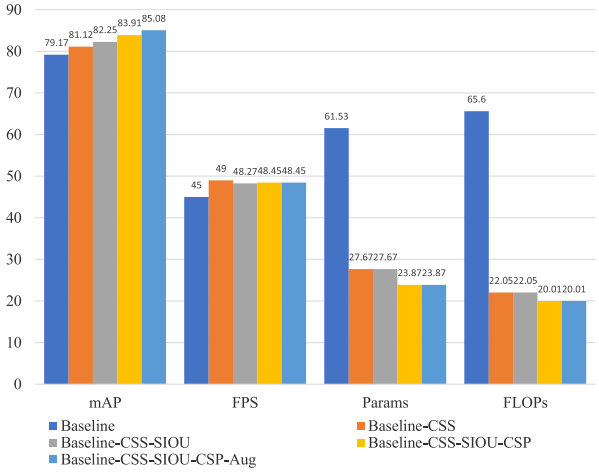


Fig. 7. Results of ablation experiments.

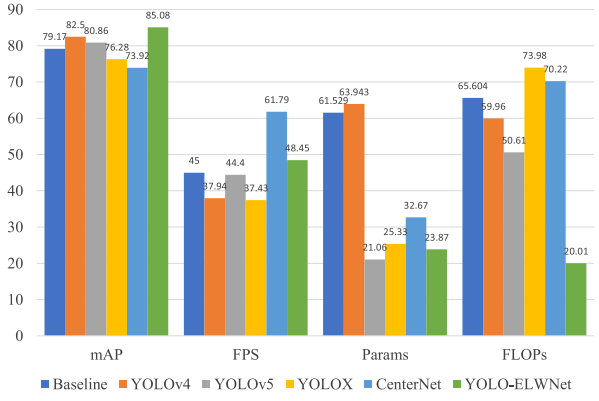


Fig. 8. Comparison of different object detectors.

results are illustrated in Table 4, where the standard YOLOv3 is adopted as the baseline. Here, CSS indicates the backbone using the CSS-CA module. SIoU represents the loss function used in the head. CSP denotes the utilization of the FPN-CSP network as the neck, and Aug represents the data augmentation with adjusted scaling ratios.

It can be seen in Table 4 that the CSS-CA-based backbone exhibits a higher mAP and FPS than the baseline. Nevertheless, the numbers of Params and FLOPs of the CSS-CA-based backbone are significantly smaller comparing with the baseline, leading to notable improvement on the object detector performance.

As shown in Table 4, by incorporating the SIoU loss function, the mAP of the Baseline+CSS+SIoU model is 82.25%, which demonstrates an additional 1.13% increase than the Baseline+CSS model. The network convergence speed is accelerated, thereby enhancing training efficiency and allowing the network to achieve optimal performance with fewer epochs.

Replacing the FPN (which is used in the neck) by the designed FPN-CSP results in 1.66% higher mAP than the Baseline+CSS+SIoU

Fig. 9. Comparison of car detection.

Fig. 10. Comparison of pedestrian detection.

model. In addition, the numbers of Params and FLOPs of the Baseline+CSS+SIoU+CSP model are slightly smaller than those of the Baseline+CSS+SIoU model, which shows that the computational burden is reduced.

Considering the limited feature extraction capability of the network, adjusting the scaling ratios (during the data augmentation phase) in the training stage yields an mAP of 85.08%, which demonstrates 5.91% higher mAP comparing with the YOLOv3 baseline. The bar chart results of the ablation study are depicted in Fig. 7.

The ablation study results indicate that each improvement can promote the network model to varying degrees, resulting in higher detection accuracy, faster detection speed, and smaller model size. The proposed method with introduced modules and modifications can effectively tackle the pedestrian and car detection problem in onboard devices.

4.3.3. Comparison of different object detectors

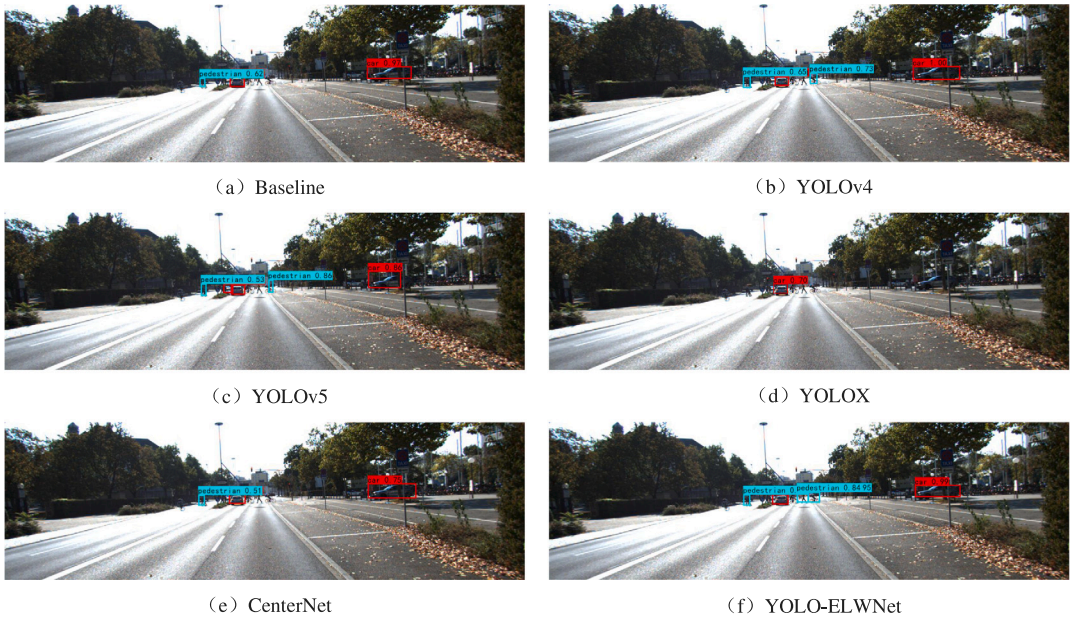
The YOLO-ELWNet method is compared with some existing popular object detection methods, including YOLOv4 [16], YOLOv5-M [39], YOLOX-M [40], and CenterNet [41]. The experimental results of the chosen object detectors are presented in Table 5.

As shown in Table 5, the YOLO-ELWNet method obtains the largest mAP comparing with the selected algorithms. Specifically, the mAP of the YOLO-ELWNet is 5.91%, 2.58%, 4.22%, 8.8%, and 11.16% higher than that of the baseline and other algorithms, respectively.

YOLOv4 achieves the second best mAP among the selected object detection methods, which seems to be another powerful detector. Nevertheless, the detection speed of YOLOv4 is inevitably affected by the additional strategies, and the relatively large model size makes it unsuitable for onboard devices. Although YOLOv5-M requires fewer model parameters than the YOLO-ELWNet, the number of FLOPs of

（a）Baseline （b）YOLOv4 （c）YOLOv5 （d）YOLOX （e）CenterNet （f）YOLO-ELWNet

Fig. 11. Comparison of close-up detection.



（a）Baseline （b）YOLOv4 （c）YOLOv5 （d）YOLOX （e）CenterNet （f）YOLO-ELWNet

Fig. 12. Comparison of remote scenery detection.

Table 5
Comparison of some currently popular object detectors.

Algorithm	Resolution	mAP (%)	FPS	Params (M)	FLOPs (G)
Baseline	416	79.17	45.00	61.53	65.60
YOLOv4	416	82.50	37.94	63.94	59.96
YOLOv5-M	640	80.86	44.40	21.06	50.61
YOLOX-M	640	76.28	37.43	25.33	73.98
CenterNet	512	73.92	61.79	32.67	70.22
YOLO-ELWNet	416	85.08	48.45	23.87	20.01

YOLOv5-M is much larger than that of the YOLO-ELWNet, which is not a suitable candidate for onboard object detection. Benefiting from the anchor-free detectors, the computational time of the CenterNet is reduced when calculating bounding boxes, demonstrating relatively faster FPS. However, the anchor-free detectors in CenterNet would lead to slower model convergence speed and lower mAP compared to other methods, which is unsuitable for onboard object detection. Abandoning the anchor-based detectors, YOLOX utilizes label assignment strategies and decoupled heads to compensate for the metric loss in mAP. Nevertheless, the performance of the YOLOX on the KITTI vision benchmark is unsatisfactory.

The selected model performance on the mAP, FPS, Params, and FLOPs are depicted in Fig. 8. Moreover, four images depicting complex scenarios are used to demonstrate the performance of the algorithms in vehicle, pedestrian, close-range, and long-range detections. Each image represents a complex scene with significant shadowing and occlusion from trees. The detection results of all selected methods are displayed in Figs. 9–12. It can be observed that CenterNet exhibits severe missed detections. YOLOv5-M has a few missed detections, and YOLOX-M

shows poor performance in capturing small objects. Notably, the proposed YOLO-ELWNet outperforms YOLOv3 and YOLOv4 in complex scenarios.

In conclusion, the proposed YOLO-ELWNet outperforms the selected popular object detectors in terms of detection accuracy, computational cost and model parameters, which demonstrates its applicability on onboard devices for real-time detection.

5. Conclusion

In this paper, an efficient lightweight neural network model, YOLO-ELWNet, has been proposed for object detection on onboard devices. The backbone block of the YOLO-ELWNet has been developed by introducing the CSS-CA module for feature extraction and the SPPF module for expanding the receptive field. The designed CSP-EB module has been embedded in the feature fusion network FPN-CSP, which effectively extracts semantic information and enhances the connectivity between feature maps. The SIoU loss function has been deployed in the head block to speed up the model convergence and improve the training efficiency. Experimental results have demonstrated that the introduced YOLO-ELWNet achieves significant improvements on the detection accuracy, detection speed, model parameters, and computational cost in comparison with some selected popular object detectors. Results have shown the effectiveness and applicability of the proposed YOLO-ELWNet for resource-constrained onboard devices.

In our future research, we will focus on exploring the model's generalization capabilities across various scenarios while optimizing it to adapt to more diverse data. Additionally, we will concentrate on decreasing the model's computational complexity to make it acceptable for devices with limited resources. Concurrently, we will pay attention to the model's real-time performance and robustness, ensuring its efficiency and stability in dynamic and uncertain environments. Through these improvements, we expect the YOLO-ELWNet model to better serve complex systems such as the automatic drive, autonomous robot, and motor fault diagnosis [42–55].

CRediT authorship contribution statement

Baoye Song: Writing – original draft, Methodology, Formal analysis, Data curation, Conceptualization. **Jianyu Chen:** Visualization, Validation, Software, Methodology, Formal analysis. **Weibo Liu:** Writing – review & editing, Supervision, Software, Project administration, Methodology, Formal analysis. **Jingzhong Fang:** Validation, Software, Investigation. **Yani Xue:** Writing – review & editing, Validation, Methodology, Formal analysis. **Xiaohui Liu:** Writing – review & editing, Supervision, Resources, Project administration, Formal analysis, Conceptualization.

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

Data will be made available on request.

References

[1] D. Pan, String stable bidirectional platooning control for heterogeneous connected automated vehicles, Int. J. Netw. Dyn. Intell. 3 (4) (2024) 100026.
[2] S. Grigorescu, B. Trasnea, T. Cocias, G. Macesanu, A survey of deep learning techniques for autonomous driving, J. Field Robot. 37 (3) (2020) 362–386.
[3] C. Katrakazas, M. Quddus, W.H. Chen, L. Deka, Real-time motion planning methods for autonomous on-road driving: state-of-the-art and future research directions, Transp. Res. Part C: Emerg. Technol. 60 (2015) 416–442.
[4] T. Luo, S. Guan, R. Yang, J. Smith, From detection to understanding: A survey on representation learning for human-object interaction, Neurocomputing 543 (2023) 126243.
[5] W. Fang, B. Shen, A. Pan, L. Zou, B. Song, A cooperative stochastic configuration network based on differential evolutionary sparrow search algorithm for prediction, Syst. Sci. Control. Eng. 12 (1) (2024) 2314481.
[6] S. Hu, J. Lu, S. Zhou, Learning regression distribution: information diffusion from template to search for visual object tracking, Int. J. Netw. Dyn. Intell. 3 (1) (2024) 100006.
[7] X. Wang, X. Kan, Z. Zhang, W. Sun, An automatic coke optical texture recognition method based on semantic segmentation model, Int. J. Netw. Dyn. Intell. 3 (4) (2024) 100022.
[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, A. Berg, SSD: Single shot multibox detector, in: Proceedings of 14th European Conference on Computer Vision, Amsterdam, Netherlands, 2016, pp. 21–37.
[9] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Seattle, USA, 2016, pp. 779–788.
[10] T.Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollar, Focal loss for dense object detection, IEEE Trans. Pattern Anal. Mach. Intell. 42 (2) (2020) 2980–2988.
[11] K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, IEEE Trans. Pattern Anal. Mach. Intell. 37 (9) (2015) 1904–1916.
[12] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Columbus, USA, 2014, pp. 580–587.
[13] R. Girshick, Fast R-CNN, in: Proceedings of IEEE International Conference on Computer Vision, Santiago, Chile, 2015, pp. 1440–1448.
[14] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2017) 1137–1149.
[15] K. He, G. Gkioxari, P. Dollar, R. Girshick, Mask r-cnn, IEEE Trans. Pattern Anal. Mach. Intell. 42 (2) (2020) 386–397.
[16] A. Bochkovskiy, C.Y. Wang, H.Y.M. Liao, YOLOv4: optimal speed and accuracy of object detection, 2020, arXiv:2004.10934.
[17] J. Redmon, A. Farhadi, YOLOv3: An incremental improvement, 2018, arXiv:1804.02767.
[18] J. Redmon, A. Farhadi, YOLO9000: Better, faster, stronger, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, USA, 2017, pp. 6517–6525.
[19] Y. Zhang, Y. Shen, J. Zhang, An improved tiny-YOLOv3 pedestrian detection algorithm, Optik 183 (2019) 17–23.
[20] B. Huang, H. Lin, Z. Hu, X. Xiang, J. Yao, An improved YOLOv3-tiny algorithm for vehicle detection in natural scences, IET Cyber- Syst. Robot. 3 (2021) 256–264.
[21] L. Lang, K. Xu, Q. Zhang, Q. Wang, Fast and accurate object detection in remote sensing images based on lightweight deep neural network, Sensors 21 (16) (2021) 5460.
[22] S. Liu, Y. Jin, Z. Ruan, Z. Ma, R. Gao, Z. Su, Real-time detection of seedling maize weeds in sustainable agriculture, Sustain. 14 (22) (2022) 15088.
[23] D. Weng, Z. Zhu, Z. Yan, M. Wu, Z. Jiang, N. Ye, Lightweight network for insulator fault detection based on improved YOLOv5, Connect. Sci. 36 (1) (2024) 2284090.
[24] Y. Wang, J. Wang, S. Zheng, M. Li, X. Xie, A railway fastener inspection method based on lightweight network, Eng. Res. Express 6 (1) (2024) 015041.
[25] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q.V. Le, H. Adam, Searching for MobileNetv3, in: Proceedings of IEEE/CVF International Conference on Computer Vision, Seoul, South Korea, 2019, pp. 1314–1324.
[26] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, Squeeze-and-excitation networks, IEEE Trans. Pattern Anal. Mach. Intell. 42 (8) (2020) 2011–2023.
[27] N.D.T. Yung, W.K. Wong, F.H. Juwono, Z.A. Sim, Safety helmet detection using deep learning: Implementation and comparative study using YOLOv5, YOLOv6, and YOLOv7, in: Proceedings of International Conference on Green Energy, Computing and Sustainable Technology, Curtin, Malaysia, 2022, pp. 164–170.
[28] P. Ramachandran, Z. Barret, Q.V. Le, Searching for activation functions, 2017, arXiv:1710.05941.
[29] N. Ma, X. Zhang, H. Zheng, J. Sun, ShuffleNetV2: Practical guidelines for efficient CNN architecture design, in: Proceedings of 15th European Conference on Computer Vision, Munich, Germany, 2018, pp. 122–138.

[30] F. Chollet, Xception: deep learning with depthwise separable convolutions, in: Proceedings of 30th IEEE/CVF Conference on Computer Vision and Pattern Recognition, Honolulu, USA, 2017, pp. 1800–1807.

[31] C.Y. Wang, H.-Y.M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, I.-Hau, Yeh, cspnet: A new backbone that can enhance learning capability of CNN, in: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, Electr Network, 2020, pp. 1571–1580.

[32] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, S. Xie, A convnet for the 2020s, in: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, USA, 2022, pp. 11966–11976.

[33] Z. Gevorgyan, Siou loss: more powerful learning for bounding box regression, 2022, arXiv:2205.12740.

[34] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The kitti vision benchmark suite, in: Proceedings of 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, USA, 2012, pp. 3354–3361.

[35] M. Everingham, S.M.A. Eslami, C.K.I. Williams, J. Winn, A. Zisserman, The PASCAL visual object classes challenge: A retrospective, Int. J. Comput. Vis. 111 (1) (2015) 98–136.

[36] D. Zhou, Q. Hou, Y. Chen, J. Feng, S. Yan, Rethinking bottleneck structure for efficient mobile network design, in: Proceedings of European Conference on Computer Vision, Glasgow, UK, 2020, pp. 680–697.

[37] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, GhostNet: more features from cheap operations, in: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, Electr Network, 2020, pp. 1577–1586.

[38] M. Tan, Q.-V. Le, EfficientNetV2: Smaller models and faster training, in: Proceedings of International Conference on Machine Learning, Virtual Conference, 2021, pp. 7102–7110.

[39] J. Glenn, YOLOv5 release v6.1, 2022, Available online: https://github.com/ultralytics/yolov5/releases/tag/v6.1 (Accessed 28 2023).

[40] Z. Ge, S. Liu, F. Wang, Z. Li, J. Sun, Yolox: exceeding yolo series in 2021, 2021, arXiv:2107.08430.

[41] X. Zhou, D. Wang, P. Krahenbuhl, Objects as points, 2019, arXiv:1904.07850.

[42] Y. Wang, C. Wen, X. Wu, Fault detection and isolation of floating wind turbine pitch system based on Kalman filter and multi-attention 1DCNN, Syst. Sci. Control. Eng. 12 (1) (2024) 2362169.

[43] H. Chen, Q. Chen, B. Shen, Y. Liu, Parameter learning of probabilistic boolean control networks with input–output data, Int. J. Netw. Dyn. Intell. 3 (1) (2024) 100005.

[44] J. Bai, Z. Yang, Z. Li, C. Shen, Y. Chen, J. Li, Trajectory tracking controller design for wheeled mobile robot with velocity and torque constraints, Int. J. Syst. Sci. 55 (14) (2024) 2825–2837.

[45] Y. Wei, Y. Wang, B. Zhu, C. Lin, D. Wu, X. Xue, R. Wang, Underwater detection: A brief survey and a new multitask dataset, Int. J. Netw. Dyn. Intell. 3 (4) (2024) 100025.

[46] C. Li, Y. Liu, M. Gao, L. Sheng, Fault-tolerant formation consensus control for time-varying multi-agent systems with stochastic communication protocol, Int. J. Netw. Dyn. Intell. 3 (1) (2024) 100004.

[47] M. Sheng, W. Ding, W. Sheng, Differential evolution with adaptive niching and reinitialisation for nonlinear equation systems, Int. J. Syst. Sci. 55 (10) (2024) 2172–2186.

[48] F. Jin, L. Ma, C. Zhao, Q. Liu, State estimation in networked control systems with a real-time transport protocol, Syst. Sci. Control. Eng. 12 (1) (2024) 2347885.

[49] B. Dai, G. Meng, Tracking control and disturbance rejection of micro quadrotor via adaptive internal model, Syst. Sci. Control. Eng. 12 (1) (2024) 2313322.

[50] J. Li, Y. Suo, S. Chai, Y. Xu, Y. Xia, Resilient and event-triggered control of singular Markov jump systems against cyber attacks, Int. J. Syst. Sci. 55 (2) (2024) 222–236.

[51] Y. Wang, C. Shen, J. Huang, H. Chen, Model-free adaptive control for unmanned surface vessels: a literature review, Syst. Sci. Control. Eng. 12 (1) (2024) 2316170.

[52] P. Gao, C. Jia, A. Zhou, Encryption-decryption-based state estimation for nonlinear complex networks subject to coupled perturbation, Syst. Sci. Control. Eng. 12 (1) (2024) 2357796.

[53] L. Fu, L. An, L. Zhang, Attitude-position obstacle avoidance of trajectory tracking control for a quadrotor UAV using barrier functions, Int. J. Syst. Sci. 55 (16) (2024) 3337–3354.

[54] G.-R. Duan, Stabilisation of four types of underactuated systems: A FAS approach, Int. J. Syst. Sci. 55 (12) (2024) 2421–2441.

[55] M. Gong, L. Sheng, D. Zhou, Robust fault-tolerant stabilisation of uncertain high-order fully actuated systems with actuator faults, Int. J. Syst. Sci. 55 (12) (2024) 2518–2530.

Baoye Song received the B.S. degree in automation in 2005, the M.S. degree in control theory and control engineering in 2008 both from Qingdao University of Science and Technology, Qingdao, China, and the Ph.D. degree in control theory and control engineering in 2011 from Shandong University, Jinan, China. From 2023 to 2024, he was a visiting Professor at Brunel University London, Uxbridge, U.K. He is currently an Associate Professor in the College of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao, China. His research interests include mobile robotics, deep learning and motor fault diagnosis.

Jianyu Chen received the B.S. degree in automation in 2021 from Weifang University, Weifang, China, and the M.S. degree in control engineering from Shandong University of Science and Technology, Qingdao, China. His research interests include object detection, deep learning and fault diagnosis.

Weibo Liu received the B.S. degree in electrical engineering from the Department of Electrical Engineering & Electronics, University of Liverpool, Liverpool, U.K., in 2015, and the Ph.D. degree in artificial intelligence in 2020 from the Department of Computer Science, Brunel University London, Uxbridge, U.K. He is currently a Lecturer in the Department of Computer Science, Brunel University London, Uxbridge, U.K. His research interests include intelligent data analysis, evolutionary computation, transfer learning, machine learning and deep learning. He serves as an Associate Editor for the Journal of Ambient Intelligence and Humanized Computing and the Journal of Cognitive Computation. He is a very active reviewer for many international journals and conferences.

Jingzhong Fang received his B.Eng. degree in automation from Shandong University of Science and Technology, Qingdao, China, in 2020, and the M.Sc. degree in data science and analytics from Brunel University London, Uxbridge, U.K., in 2021. He is currently pursuing the Ph.D. degree in Computer Science at Brunel University London, Uxbridge, U.K. His research interests include data analysis and deep learning techniques.

Yani Xue received her Ph.D. degree in Computer Science from Brunel University London in 2021. She is currently a Lecturer at Brunel University London. Her main research interests include multi/many-objective optimization, evolutionary computation, search-based software engineering, engineering applications, simulation, and high-performance data analytics.

Xiaohui Liu received his B.Eng. Degree in Computing from Hohai University, Nanjing, China, in 1982 and his Ph.D. in Computer Science from Heriot-Watt University, Edinburgh, UK, in 1988. He is currently a Professor of Computing at Brunel University of London, specializing in research on Artificial Intelligence and Intelligent Data Analysis. His work spans a range of applications, including biomedicine and engineering.