electronics

MDPI

Article

Object Detection Network Based on Module Stack and Attention Mechanism

Xinke Dou [1,2,3], Ting Wang [2,3,*], Shiliang Shao [2,3,*] and Xianqing Cao [1]

[1] School of Information Engineering, Shenyang University of Chemical Technology, No. 11, 11th St., Tiexi District, Shenyang 110142, China; douxinke@sia.cn (X.D.); cxq_sy@163.com (X.C.)
[2] State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, No. 114, Nanta St., Shenhe District, Shenyang 110016, China
[3] Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, No. 135, Chuangxin Rd., Hunnan District, Shenyang 110003, China
* Correspondence: wangting@sia.cn (T.W.); shaoshiliang@sia.cn (S.S.)

Abstract: Currently, visual computer applications based on convolutional neural networks are rapidly developing. However, several problems remain: (1) high-quality graphics processing equipment is needed, and (2) the trained network model has several unnecessary convolution operations. These problems result in a single-stage target detection network that often requires unnecessary computing power and is difficult to apply to equipment with insufficient computing resources. To solve these problems, based on YOLOv5, a YOLOv5-L (YOLOv5 Lightweight) network structure is proposed. This network is improved using YOLOv5. First, to enhance the inference speed of the detector on the CPU, the PP-LCNet (PaddlePaddle-Lightweight CPU Net) is employed as the backbone network. Second, the focus module is removed, and the end convolution module in the head network is replaced by a deep separable convolution module, which eliminates redundant operations and reduces the amount of computation. The experimental results show that YOLOv5-L enables a 48% reduction in model parameters and computation compared to YOLOv5, a 35% increase in operation speed, and a less than 2% reduction in accuracy, which is significant in the environment of low-performance computing equipment.

Keywords: lightweight; YOLOv5; PP-LCNet; object detection; attention mechanism

check for updates

Citation: Dou, X.; Wang, T.; Shao, S.; Cao, X. Object Detection Network Based on Module Stack and Attention Mechanism. *Electronics* **2023**, *12*, 3542. https://doi.org/10.3390/electronics12173542

Academic Editor: Savvas A. Chatzichristofis

Received: 29 July 2023
Revised: 14 August 2023
Accepted: 16 August 2023
Published: 22 August 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the rapid advancement of technology, visual sensors have become ubiquitous in all aspects of human life, enabling the detection of pedestrians, animals, and objects. Target detection tasks typically involve classifying and recognizing objects using markers and utilizing bounding boxes for localization. Currently, an increasing number of target detection applications, such as face recognition, vehicle detection, and smart home systems, are being applied to resource-constrained edge devices. As object recognition continues to evolve, visual sensors are generating vast amounts of data, and the most effective solution to this challenge is deep learning. Before the introduction of the AlexNet [1] model in 2012, traditional image processing techniques were commonly used for object detection. In industrial production, there are usually tasks such as defect detection and object detection. The utilization of robots instead of manual labor not only reduces labor costs, but also enhances production efficiency, thus fostering industrial advancement. Similarly, agriculture and the military are pivotal domains in object detection. For instance, drones equipped with visual devices can perform tasks such as crop detection and gathering advanced weather information. The same applies to military applications as well. These examples highlight the immense potential of object detection applications. However, owing to the constraints imposed by specific tasks and the high costs associated with the related equipment, the full implementation of these products remains challenging. Consequently,

developing a deep-learning network model that can perform real-time detection on limited devices is of great significance.

Currently, two methods are available for deploying object-detection models on restricted devices. The first involves compressing models or data through specific operations [2], resulting in lightweight models. The second approach involves the distribution of the model across multiple devices for distributed computing [3]. Ref. [4] discussed methods that can handle deep learning in distributed environments, such as federated learning and transfer learning. The transfer learning method based on the VisDrone dataset not only solves the problem of limited data, but also outperforms various well-known object detection methods in terms of results. The primary research challenge for the former method lies in striking the correct balance between model accuracy and detection speed, whereas the latter method raises concerns regarding privacy, data security, and additional costs associated with debugging multiple devices. Considering the advantages and disadvantages of these two approaches, a lightweight model is used.

The two-stage detector [5] first becomes a candidate detection region, and then identifies the region with a relatively high probability based on the region category as the final target detection region. The one-stage detector [6] skips the first step and directly prompts the possibility of the target category and the pixel coordinates of the image where the target is located. Common representative methods of two-stage detection include the R-CNN [7], Fast R-CNN [8], and Faster R-CNN [9], whereas representative methods of one-stage detection include the You Only Look Once (YOLO) and Single Shot Multibox Detector (SSD) [10] series. The YOLO [11] algorithm was first proposed by Joseph Redmon in 2016 and was repeatedly updated by the author, introducing feature pyramid networks (FPN) and a better basic classification network, Darkenet-53. The classifier also changes from a normalized exponential function (softmax) to a binary cross-entropy loss [12]. YOLOv3 was proposed in 2018 [13]. Later, Bochkovskiy proposed YOLOv4 in 2020 [14], whereas in the same year, Ultratics proposed YOLOv5, which is only 27 MB. Compared with the 244 MB size of YOLOv4, the size of YOLOv5 was reduced by approximately nine times. With the progress made in backbone network architecture, many researchers have begun to integrate these networks into their target detection models.

One-stage detection methods are typically faster than two-stage detection methods [15]. Current one-stage detection methods typically construct lightweight models by reducing their depth and width. However, although the detection speed can be improved using this scheme, a relatively large drop in detection accuracy is almost unacceptable. Therefore, researchers are constantly exploring and designing simplified structures and lightweight modules to improve detection speed while maintaining as much accuracy as possible.

The current study first combines the lightweight network model PP-LCNet [16] with YOLOv5, optimizes and prunes many redundant calculations in the original network, and constructs a lightweight backbone network. Combined with the ShuffleNetv2 [17] network structure strategy, the backbone network was further improved, and PP-LCNet was reconstructed by stacking deep-separable modules with an even number of convolution cores of different sizes. A convolutional block attention module (CBAM) was applied to the neck network, and ablation experiments were performed on the neck network to verify the effect of PP-LCNet, replacing a larger convolution at the end of the network for accuracy and speed. This satisfies the demands for real-time target detection in low-performance devices.

The main contributions of this study are as follows: (1) A new lightweight model was proposed for target detection, which is helpful for achieving real-time target detection on low-performance devices. (2) The backbone and neck networks were carefully designed to balance the accuracy loss caused by an increase in the network speed. (3) The characteristics of current mainstream lightweight networks were summarized, such as equal channel widths (equal channels for input and output feature matrices), avoiding large numbers of group convolutions and network branches, and integrating them into network models.

The remainder of this article is organized as follows: Section 2 introduces relevant work on lightweight models. Section 3 introduces the basic architecture of the YOLOv5

network model and the improvements made to it. Section 4 presents the experimental results and a discussion of the experimental results. Finally, Section 5 gives the conclusion of this paper.

2. Related Work

In this section, we briefly introduce the recent methodologies related to lightweight object detection models, including model compression, depth separation/model scaling, and hybrid approaches.

2.1. Model Compression

Compression, quantization, pruning, and knowledge distillation are several commonly used methods for compressing models. Model compression focuses on designing models more efficiently with minimal performance loss, with the aim of reducing the computational redundancy caused by large models. For instance, the compressed histogram method proposed in [18] enhances the image quality without increasing the computational burden, yielding good results. In [19], a PG-YOLO single-stage network was introduced, utilizing multimodal blocks (PM) to learn object features at multiple scales and shapes. Furthermore, a group-attention module (GAM) was devised to better utilize the diverse connectivity features in the network, resulting in model compression and parameter reduction. Pruning is a straightforward technique for model compression that involves the removal of weights, neurons, and other components from an original large training network. Quantization, on the other hand, is a low-level but effective method for model compression. Numerous researchers have dedicated their efforts to these techniques, seeking to achieve compact models without sacrificing performance [20]. He et al. [21] proposed an iterative, two-step algorithm that efficiently pruned each layer of the network structure. This approach effectively reduces cumulative errors and can be extended to other widely used backbone networks, such as ResNet and VGGNet. Liu et al. [22] proposed an extremely lightweight network. SAMNet Quantization is another method for reducing model size. Quantification typically reduces model size by storing the model in different formats during training or model deployment. The former is referred to as quantization-aware training, whereas the latter is known as post-training quantization. Knowledge distillation is a commonly employed technique to achieve lightweight models. This involves learning the representation of a large teacher model from a smaller student model, effectively transferring knowledge, and reducing its complexity.

2.2. Depth Separation/Model Scaling

MobileNet [23], ShuffleNet [24], and others have used deep separable convolution to reduce the amount of computation and design efficient feature extraction modules. These separable convolutional structures can not only achieve spatial separation but also deep separation; that is, they can use spatial separation to replace a large convolutional kernel with several small convolutional kernels. The advantage of doing so is that it can reduce computational complexity, generate fewer parameters, and enable the network to process data more efficiently. Currently, many researchers are committed to building a network of depth separable convolutional modules stacked to create high-quality, lightweight architectures. FeatherCNN [25] proposed a joint layer-by-layer optimization structure that utilizes layer fusion to remove redundant computing, which, to some extent, solves the upper limit problem of mobile device performance. MiniNet [26] proposed an extremely lightweight deep learning-based model consisting of DepthNet and PoseNet. DepthNet is responsible for inputting a single still image. Lite Depthwise Net [27] proposed a new network architecture for hyperspectral classification based on three-dimensional deep convolution, using deep convolution and point-by-point convolution, which can achieve high classification performance with minimal parameters.

2.3. Hybrid Approaches

Currently, the most successfully designed lightweight models employ various hybrid methods. Different module designs, such as ghost modules, are used to change the original BN. The existing successful cases are not limited to a single model but attempt to combine the advantages of multiple strategies. Some researchers have followed the current trends and modified them by combining YOLO, SSD, and other architectures to create specific models suitable for new applications. The authors of [28] proposed a lightweight directional object detector for RSOD tasks. The backbone uses MobileNetv2, which employs a channel separation aggregation module that can simplify stacking and separable convolution complexity and a dynamic receptive field module that can improve the performance of feature extraction by dynamically customizing the convolution core and its perception range. The authors of [29] proposed a pedestrian detection method based on data augmentation that combines a lightweight CNN (LCNN)-based pedestrian recognition model with YOLOv3 to enlarge the image size of pedestrians smaller than the specified value. This end-to-end framework improves detection stability and accuracy. In [30], a lightweight model based on a YOLOv3 tiny network was proposed, which incorporates spatial pyramid pooling into the network. The network consists of pooling layers with different proportions, which are used to connect features to enhance network learning ability. The authors of [31] introduced MobileNetv3 into the backbone network, and a simple and fast cross-scale connection feature pyramid network was proposed, which not only improved the model's detection speed, but also improved the model's ability to detect small targets. In [32], MobileNetv2 was combined with YOLOv3, and a Gaussian model was introduced to improve detection performance. The authors of [33] proposed a multi-objective NAS framework for edge devices that improves scalability and practicality using an improved MobileNetv2 search space. In [34], a single-stage detection neural network was used, and Tiny YOLO and MobileNetv2-SSD were proposed by simplifying the backbone network structure.

2.4. Other Detection Works

As we all know, with the development of technology, more and more sensors are being applied to target detection tasks in various scenarios in daily life, such as LiDAR and global positioning systems [35]. Our lightweight network model has great potential for application. The authors of [36] proposed a data acquisition and analysis platform for vehicle trajectory extraction, reconstruction, and evaluation of auto drive systems (ADS) based on the collaborative perception of the Internet of Vehicles and using the LiDAR information target detection algorithm based on deep learning. The authors of [37] used RBG cameras to apply YOLOv4 to the target detection task of tram autonomous driving [38]. By using radar and cameras for data fusion, robust continuous multi-target tracking can be achieved, which can be applied to design safer and more efficient intelligent driving systems.

Coincidentally, there are many other detection techniques that need to be applied to lightweight network models. In [39], by using ordinary detectors to treat infrared images as grayscale visible light images to provide primary detection results and then converting infrared images into corresponding temperature maps for detection, pedestrian detection is carried out through this two-stage combination. The authors of [40] used cascaded classifiers for vehicle detection tasks and used them on embedded platforms.

In this study, we established a lightweight object-detection network that can quickly detect performance-limited devices. In addition, we combined it with the improved object detection algorithm YOLOv5, which has a significantly improved detection speed with minimal loss of detection accuracy.

3. Materials and Methods

In this section, we first describe the network architecture of YOLOv5. We then introduce our improved network architecture based on the two parts of the backbone and neck networks.

3.1. YOLOv5

The backbone network of YOLOv5 [41] is primarily composed of a focus module, a cross-stage partial network (CSPNet) module, and a spatial pyramid pooling (SPP) module. The backbone network uses several residual modules, whereas the neck uses common convolution modules. Similar to YOLOv4, it uses a feature pyramid structure and a pixel aggregation network (PAN). The head network is primarily used for the final detection by applying an anchor frame to the feature map and generating the final output vector with category probability, confidence, and other information. The loss function includes classification, localization, and confidence loss. To satisfy the detection speed requirements of this study, YOLOv5s with the minimum depth and width of the model was selected as the reference network, and the network was modified and verified using v5.0.

The input end of YOLOv5 is the same as that of YOLOv4. CutMix adopts the mosaic data enhancement method and re-splices four pictures by random scaling, cutting, and arranging, thereby reducing the pressure on the computing equipment and enriching the dataset. The input terminal of YOLOv5 includes an adaptive anchor-frame calculation and image-scaling strategy based on a k-means clustering algorithm [42].

The focus structure shown in Figure 1 was proposed for the backbone network. The slicing operation of this structure, which is essentially a downsampling operation, is a key part. YOLOv5 replaces the original BottleneckCSP module with the C3 module in v5.0. BottleneckCSP is divided into two forms, residual and non-residual, which are applied to the backbone and neck networks, respectively. The main difference between the C3 module and the BottleneckCSP module is that convolution is missing.

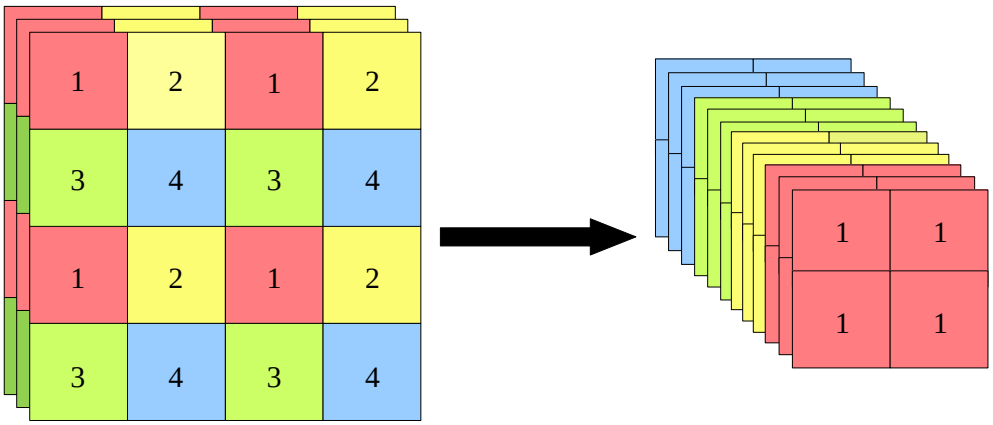


Figure 1. Focus slice operation.

The neck network uses an FPN + PAN structure. FPN transmits information from high dimension to low dimension, making the big target more accurate. PAN transmits information from low to high dimensions, making small targets more accurate.

The prediction part uses the binary cross-entropy loss function to calculate the category probability and target confidence score loss and the generalized intersection and union ratio loss (GIoU loss) to calculate the target box regression loss.

3.2. YOLOv5-L

In this study, the YOLOv5 model was lightweight and improved by the backbone and neck networks. In the backbone network part, PP-LCNet is used to replace the original backbone network, and the network structure of PP-LCNet itself is reconstructed using the stacking strategy of ShuffleNetv2's even-times depth convolution module plus a point-by-point convolution module, that is, a depth-separable module. In the neck network, CBAM [43] is added to the convolution module at the front end so that the network model can better obtain image features. The end convolution module was replaced by a depth-separable convolution module in which a 5×5 convolution kernel replaced the original

3 × 3 convolution kernel and added squeeze and excitation (SE) modules. The overall structure of YOLOv5-L is shown in Figure 2.

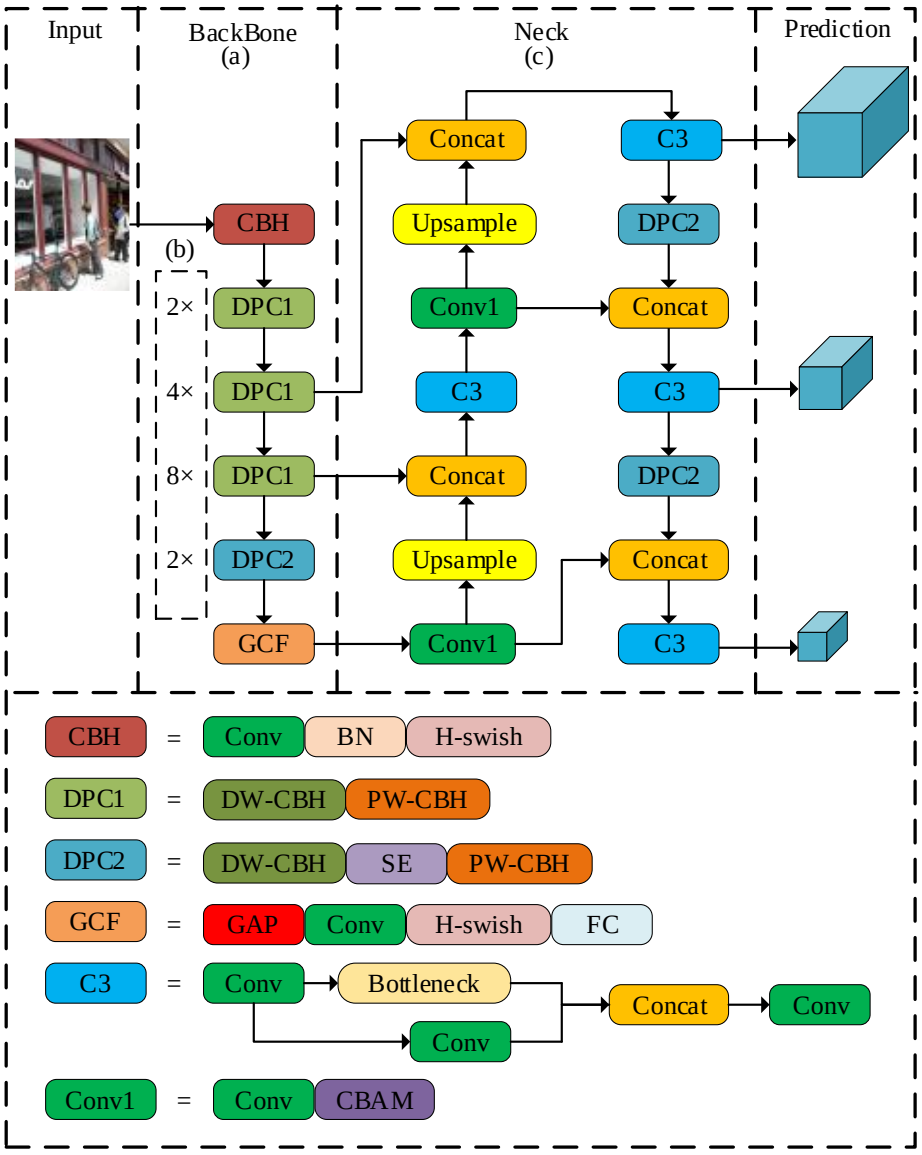


Figure 2. YOLOv5-L network structure. (**a**) Backbone network upgrade: Replace YOLOv5's original backbone network with a lightweight network, PP-LCNet. (**b**) Backbone network optimization: Reconstruct PP-LCNet using the stacking strategy of 2, 4, 8, and 2 even-fold modules used in ShuffleNetv2. (**c**) Improvement of neck end network: Introduce CBAM attention mechanism; use PP-LCNet end depth detachable module.

3.2.1. Improvements in Backbone Networks

This study is dedicated to improving the reasoning speed of a target detection network on the CPU side. The original YOLOv5 target detection model was improved, and PP-LCNet was combined to improve the reasoning speed. The main improvements to the backbone network include: (1) Using the PP-LCNet to replace the original backbone network of YOLOv5, which makes the entire network lightweight. (2) Inspired by the stacking strategy of 2, 4, 8, and 2 even-time modules used by ShuffleNetv2, the PP-LCNet module was reconstructed to improve detection accuracy.

PP-LCNet is a lightweight CPU model based on MKLDNN acceleration. In image classification tasks, PP-LCNet is significantly better than ShuffleNetv2 and MobileNetv3 in terms of reasoning delay-precision balance. This network model uses the DepSepConv

module in MobileNetv1 as the basic module, without feature fusion (concat and add) operations. The basic PP-LCNet network is constructed by stacking the DepSepConv module and adding a squeeze excitation module. The structure of the PP-LCNet is shown in Figure 3.

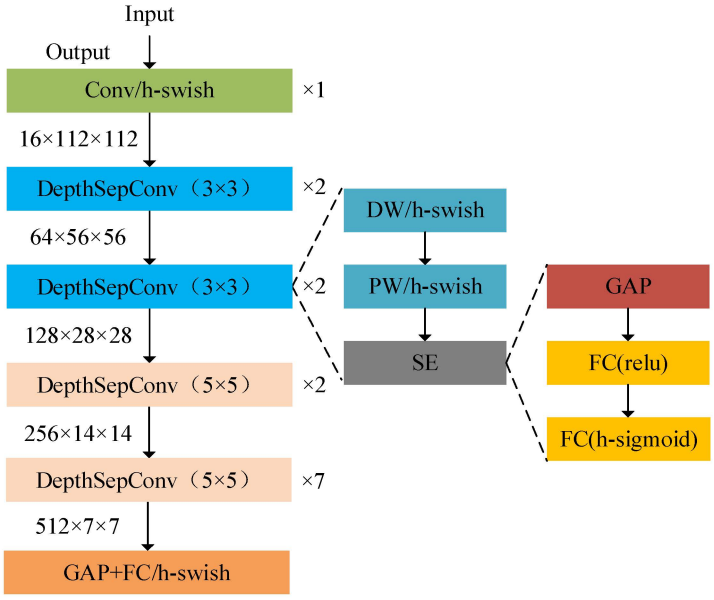


Figure 3. PP-LCNet network structure.

Simultaneously, the H-swish activation function was adopted. The activation function runs on the artificial neural network. It is used to map the input of the neurons to the output. An effective activation function is important for learning and understanding the complex nonlinear functions of a network model. When an activation function is used, the neuron introduces nonlinear factors, such that the network can arbitrarily approach any nonlinear function. Therefore, the proposed network model can be applied to other nonlinear models. EfficientNet [44] was proposed to introduce the swish function into the network, and it achieved better results. This function possesses attributes such as being unbounded, having a smooth and non-monotonic nature, and outperforming ReLU in deep models. To address this issue, the sigmoid function calculation involves an exponential operation and high computational complexity; MobileNetv3 changes the swish function and proposes an H-swish function. ReLU6 was selected as the approximate function for two reasons: (1) Almost all software and hardware frameworks support ReLU6. (2) The possible precision loss caused by the approximate sigmoid function is eliminated. Therefore, the activation function was introduced into YOLOv5-L, which significantly improved the network performance, while the inference time was almost unchanged. The expressions for the H-swish and sigmoid activation functions are shown in Equations (1) and (2), respectively. The H-swish and sigmoid approximation functions ReLU6 used in YOLOv5-L are expressed in Equations (3) and (4), respectively.

$$f(x) = x \cdot sigmoid(\beta x) \tag{1}$$

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

$$H - swish(x) = x\frac{ReLU6(x+3)}{6} \tag{3}$$

$$ReLU6(x) = \min(\max(0, x), 6) \tag{4}$$

ShuffleNetv2 studies two indicators of computing complexity, FLOPs and running speed, commonly used in the performance evaluation standards of small networks in terms of lightweight and summarizes several efficient improvement strategies: (1) An equal channel width can reduce storage access costs. (2) A large number of packet convolutions increases storage access. (3) Too many network branches reduce parallelization speed. (4) Element-level operations that cannot be ignored. Based on the above ideas, ShuffleNetv2 constructs a network structure in the form of 2, 4, 8, and 4 evenly stacked convolution modules. Therefore, the deep separable convolution module of the PP-LCNet stack was reconstructed based on this stacking module. The final network structure adds only three layers, resulting in a lower cost of parameters and computation. The reconstructed PP-LCNet network structure is shown in Figure 4.

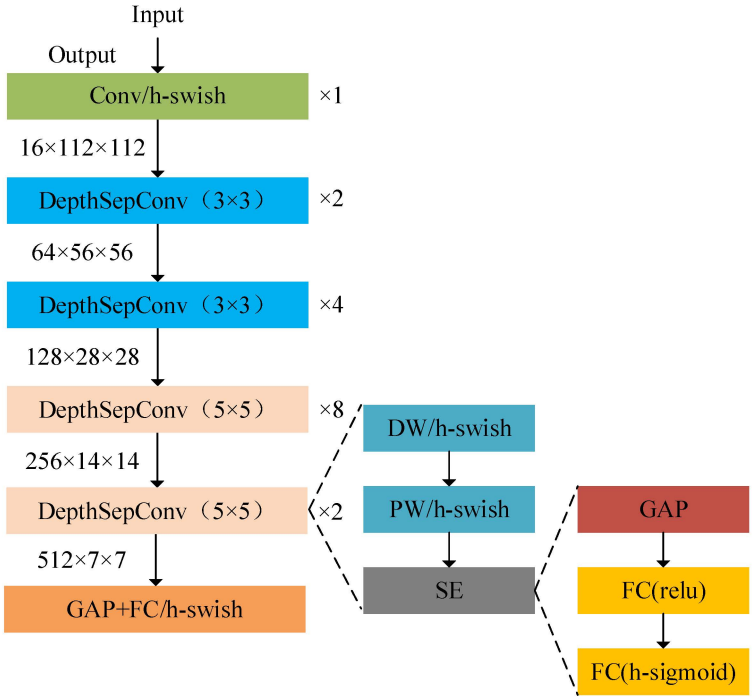


Figure 4. Improving the PP-LCNet network structure.

3.2.2. Improvements in Neck Networks

The improvement in the neck network mainly includes the following: (1) Adding the CBAM to improve detection accuracy. (2) The neck network end was replaced with a deep separable convolution network, and a larger convolution core was used to improve the network detection speed and reduce the network complexity.

The CBAM is a simple and effective attention module that belongs to feedforward convolutional neural networks and is divided into spatial and channel attention modules. It can be seen from Figure 5 that the channel attention module (CAM) is in front, and the spatial attention module (SAM) is behind.

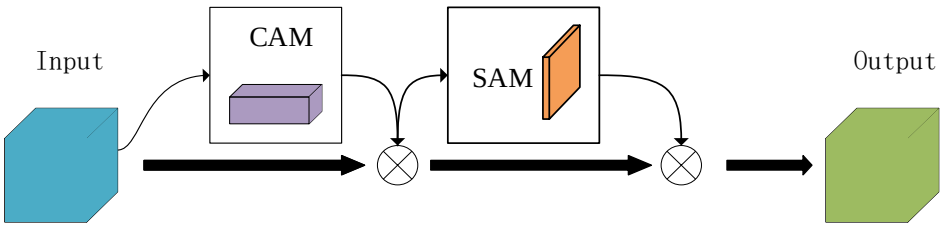


Figure 5. CBAM attention mechanism.

After inputting the feature map, the CAM module is applied first. It performs global average pooling (GAP) and global maximum pooling (GMP) [45] based on the width and height of the feature map. The attention weights are then normalized using the sigmoid function. Finally, the original input channels are weighted channel by channel through multiplication with the normalized attention weights.

After extracting the channel features, GMP and GAP are applied based on the width and height of the input image, resulting in a dimensional change from H × W to 1 × 1, followed by 7 × 7. In the spatial attention module, GAP and GMP were used to obtain the spatial attention features. Two convolutions establish correlations between the spatial features while maintaining the input and output dimensions. A convolution operation with a 7 × 7 kernel reduces the number of parameters and computations, facilitating the establishment of high-dimensional spatial feature correlations. After applying CBAM, the new feature map incorporates both channel and spatial dimension information. This significantly improves the connectivity of each feature in terms of channel and space, making it more effective for extracting the target features [46].

The PP-LCNet was verified using ablation experiments. By using a larger convolution core at the end of the network model, the network detection speed can be improved, and the network computing complexity can be reduced. Therefore, considering the large amount of computation and slow reasoning speed of the traditional convolution module, the end convolution module at the neck of the original YOLOv5 network structure was changed into a deep separable convolution module, and the SE module was added to it to reduce the amount of computation and increase the reasoning speed. Finally, the results were verified through an ablation experiment.

4. Results and Discussion

In this section, we experiment with the collected header dataset and the compressed MS COCO dataset (Minico Dateset), and compare the proposed model with other methods. We designed an ablation study to investigate the performance of the improved module.

4.1. Experimental Environment and Parameter Setting

The machine operating system for model training was Windows10, the CPU was i5-7300HQ @ 2.50 GHz, the memory size was 16 GB, the GPU model was NVIDIA GeForce GTX 1050 Ti, the memory size of the graphics card was 4 GB, and the in-depth learning framework were Python 1.7.1, cuda11.7, and cuDNN8.0. The machine operating system used in the reasoning experiment was Windows10, the CPU was i7-8650U @ 1.90 GHz, 16 GB of memory, and there was no GPU device.

The image input size in this experiment is 640 × 640 pixels, and the batch size is affected by the GPU device. If the setting is too small, the gradient may change, making it difficult for the network to converge. Through multiple training, we selected the batch size as 16 with the highest GPU utilization rate and set the number of epochs to 100. The selection of an appropriate anchor box accelerates the regression of the bounding box. There is an adaptive anchor box selection function in YOLOv5. According to the provisions of YOLOv5, if the best possible recall (BPR) is greater than or equal to 0.98, the default anchor box size of YOLOv5 is used. Otherwise, the k-means and genetic learning algorithms were used to recalculate the anchor box conforming to this dataset. Because the BPR in this experiment was one, the YOLOv5 default anchor box was selected.

4.2. Data Set

To verify the universality of our proposed method, two datasets were used in the experiment. One of them was extracted from 4360 portrait images and classroom surveillance images collected online, which were only classified as heads. The label annotation tool is used for label annotation. Another dataset is a larger and more complex COCO dataset, while considering the issue of insufficient device performance. Compressing the COCO dataset and extracting a portion of the data (extracting 1 W images using random sampling)

to create the Minico dataset. The partition ratio of the training set, validation set, and test set for both datasets is 8:1:1.

The evaluation indicators use Precision (P), Recall (R), Average Precision (AP), and the average Average Precision (mAP) of all categories of AP values as evaluation indicators, where mAP is divided into mAP@0.5 and indicates the mAP when the IoU threshold is set to 0.5, and mAP@0.5:0.95 represents mAP under different IoU thresholds (0.5 to 0.95, in steps of 0.05), which were also selected as evaluation indicators in this experiment. The calculation formula for each indicator is

$$P = \frac{T_P}{T_P + F_P} \times 100\% \tag{5}$$

$$R = \frac{T_P}{T_P + F_N} \times 100\% \tag{6}$$

$$AP = \frac{\sum_{\mathrm{m}} P}{n} \tag{7}$$

$$\mathrm{m}AP = \frac{\sum_{i=1}^{m} AP}{N} \tag{8}$$

where true positive (T_P) represents cases where both the sample and the prediction are true; false positive (F_P) represents cases where the sample is false, but the prediction is true; true negative (T_N) represents cases where both the sample and the prediction are false; and false negative (F_N) represents cases where the sample is true, but the prediction is false.

4.3. Evaluation Criterion

This part of the study used the ablation experimental method to verify the performance improvement and change in the improved model compared with the original YOLOv5 model. For the CPU device, only 10% of the dataset was tested. The parameter (Params), Giga Floating-point Operations Per Second (GFLOPS), mAP@0.5, mAP@0.5:0.95, calculate time, and other evaluation indicators were used to analyze the training and test results.

First, test the performance of YOLOv5's model, then replace the backbone network of YOLOv5 with the PP-LCNet network structure for testing, then test the deep separable convolution module in the PP-LCNet network in the backbone network according to the ShuffleNetv2 network structure strategy for 2, 4, 8, and 2 even-time stacking, and finally, add the first two layers in the neck network to the CBAM attention mechanism. The convolution module of the last two layers is changed to PP-LCNet's deep separable convolution module, and the SE module is used in it. The final performance results are listed in Tables 1 and 2. YOLOv5 represents the original YOLOv5's network model; PP-LCNet represents the network model for replacing the YOLOv5 backbone network with PP-LCNet; LC-Shuffle represents the network model for improving PP-LCNet according to the module stacking strategy of ShuffleNetv2; and YOLOv5-L represents the final network model for improving the neck network based on LC-Shuffle. The up arrow indicates an improvement compared to the previous improvement, while the down arrow indicates a decrease compared to the previous improvement (The arrows in Tables 3 and 4 have the same meanings).

Table 1. Performance comparison of different models (Head dateset).

Model	mAP@0.5	mAP@0.5:0.95	Param(M)	GFLPs	Calculate Time (ms)
YOLOv5s	0.943	0.481	7.06	16.5	347
PP-LCNet	0.917↓	0.425↑	4.13↑	8.5↑	242↑
LC-Shuffle	0.925↑	0.440↑	4.31↓	9.3↓	280↓
YOLOv5-L (Ours)	0.926↑	0.446↑	3.70↑	8.6↑	216↑

Table 2. Performance comparison of different models (Minicoco dateset).

Model	mAP@0.5	mAP@0.5:0.95	Param(M)	GFLPs	Calculate Time (ms)
YOLOv5s	0.449	0.281	7.23	17	355
PP-LCNet	0.198↓	0.162↑	4.33↑	9.1↑	246↑
LC-Shuffle	0.412↑	0.252↑	4.52↓	10.0↓	273↓
YOLOv5-L (Ours)	0.423↑	0.256↑	3.93↑	9.2↑	211↑

According to the data analysis presented in Table 1, after replacing the YOLOv5 backbone network with PP-LCNet, the number of model parameters decreased by approximately 41%, the number of calculations decreased by 48%, the reasoning speed increased by 30%, and the accuracy decreased by approximately 3%. After replacing the backbone network with the reconstructed PP-LCNet, the accuracy increased by 1.6%, the number of model parameters increased by 12%, the number of calculations increased by 6%, and the reasoning speed decreased by 11%. After adding the neck network improvement strategy, compared with the original PP-LCNet, the number of model parameters was reduced by 10%, the number of calculations was increased by only 1%, and mAP@0.5:0.95 improved by 2.1%. Compared with the YOLOv5 network model, it is only reduced by 1.7%, but the speed is increased by 35%, and the number of model parameters and calculations is reduced by nearly 50%.

In Table 2, due to the increase in data categories and the decrease in the amount of data for each category, the overall accuracy has decreased significantly. However, the number of parameters and calculations has still been greatly reduced, and the inference speed has also been greatly improved, with only a decrease of about 2.5% in accuracy.

Based on the above results, the lightweight improvement made in this study can greatly improve the reasoning speed of low-performance equipment while ensuring as much accuracy as possible. And it has a certain degree of universality.

4.4. Comparative Experiment

To better verify the effect of the improved YOLOv5-L network model, mainstream lightweight network models such as MobileNetv3 and ShuffleNetv2 were compared in this module. In the same experimental machine environment, an identical dataset was employed for training and testing to compare the performance of the network model. The evaluation index was the same as that used in the ablation experiments. According to the analysis in Table 3, the parameters of the improved network model proposed in this paper are 4% more than MobileNetv3 and 140% more than ShuffleNetv2. The GFLOPs of the improved network model proposed in this paper are 36% more than MobileNetv3, 132% more than ShuffleNetv2, but the accuracy is improved by about 5%, and the reasoning speed is faster than MobileNetv3 and ShuffleNetv2.

Table 3. Performance comparison of lightweight network model (Head dateset).

Model	mAP@0.5	mAP@0.5:0.95	Param(M)	GFLPs	Calculate Time (ms)
MobileNetv3	0.879	0.388	3.54	6.3	232
ShuffleNetv2	0.876	0.387	1.54	3.7	221
YOLOv5-L (Ours)	0.926↑	0.446↑	3.70↑	8.6↑	216↑

According to the analysis in Table 4, the parameters of the improved network model proposed in this paper are 5% more than MobileNetv3 and 138% more than ShuffleNetv2. The GFLOPs of the improved network model proposed in this paper are 31.4% more than MobileNetv3, 130% more than ShuffleNetv2, but the accuracy is improved by about 18% and 20%, and the reasoning speed is faster than MobileNetv3 and ShuffleNetv2.

Table 4. Performance comparison of lightweight network model (Minicoco dateset).

Model	mAP@0.5	mAP@0.5:0.95	Param(M)	GFLPs	Calculate Time (ms)
MobileNetv3	0.242	0.127	3.75	7.0	215
ShuffleNetv2	0.225	0.116	1.65	4.0	212
YOLOv5-L (Ours)	0.423↑	0.256↑	3.93	9.2	211↑

Compared with MobileNetv3 and ShuffleNetv2, the YOLOv5-L model's parameter quantity and calculation quantity are improved, but most of the low-performance devices do not meet the requirements of the model's parameter quantity and calculation quantity The reasoning speed of the model is faster than these two network models, and the accuracy is increased. Further, the accuracy of multi-classification has been greatly improved. As a result, the YOLOv5-L network model proposed in this study exhibited a remarkably high cost–performance ratio.

In Figure 6, we present the training process of the improved network on the Minicoco dataset. In Figure 7, we also provide some visual examples of detection outputs from additional algorithms on the MS COCO test dataset. The detection results from YOLOv5-L, Mobilenetv3, and Shufflenetv2 are displayed in the first to third columns, respectively. In the detection comparison of the first line of pictures, Mobilenet and Shufflenet networks have some missed detections, such as couch and book categories. In the second line of detection comparison, the Mobilenet and Shufflenet networks missed a certain number of cars and identified the bus category as a train. In the third line of detection comparison, Mobilenet and Shufflenet networks mistakenly detected decorations similar to watches as clocks. In the fourth line of detection comparison, YOLOv5-L mistakenly detected the truck as a bus, while Mobilent detected the truck but missed the inspection of the handbag and car, and the Shufflenet network missed the inspection of the handbag and truck. According to the results, our method can capture contextual information more accurately and detect it more comprehensively.

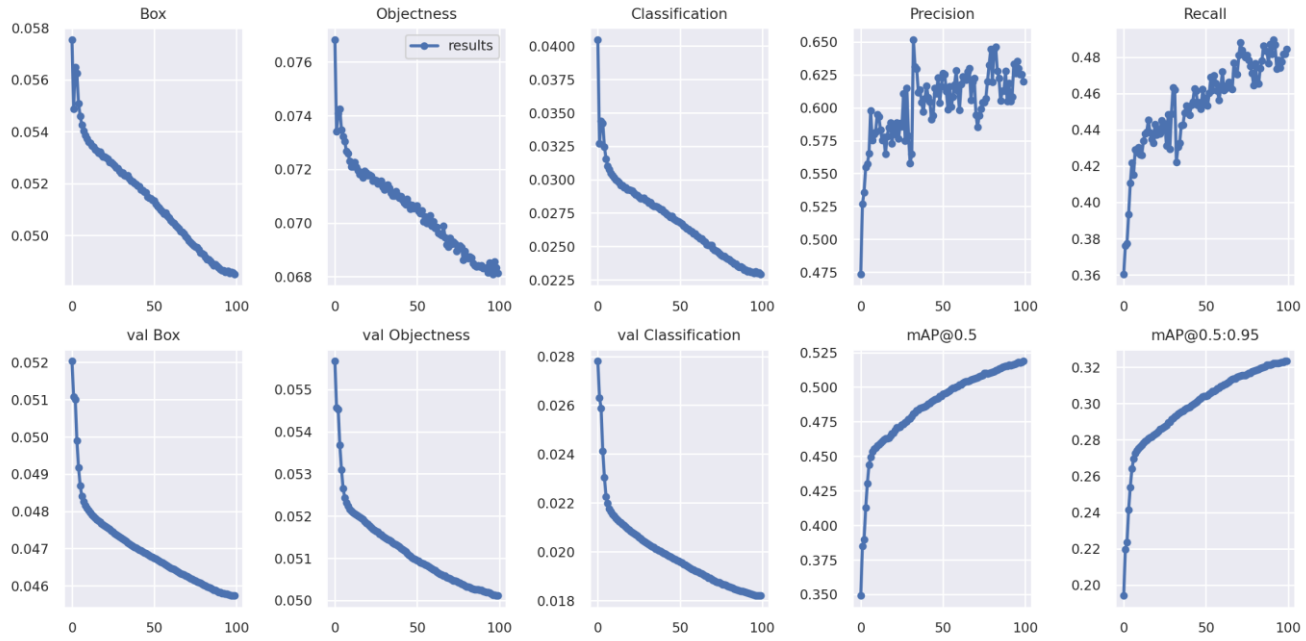
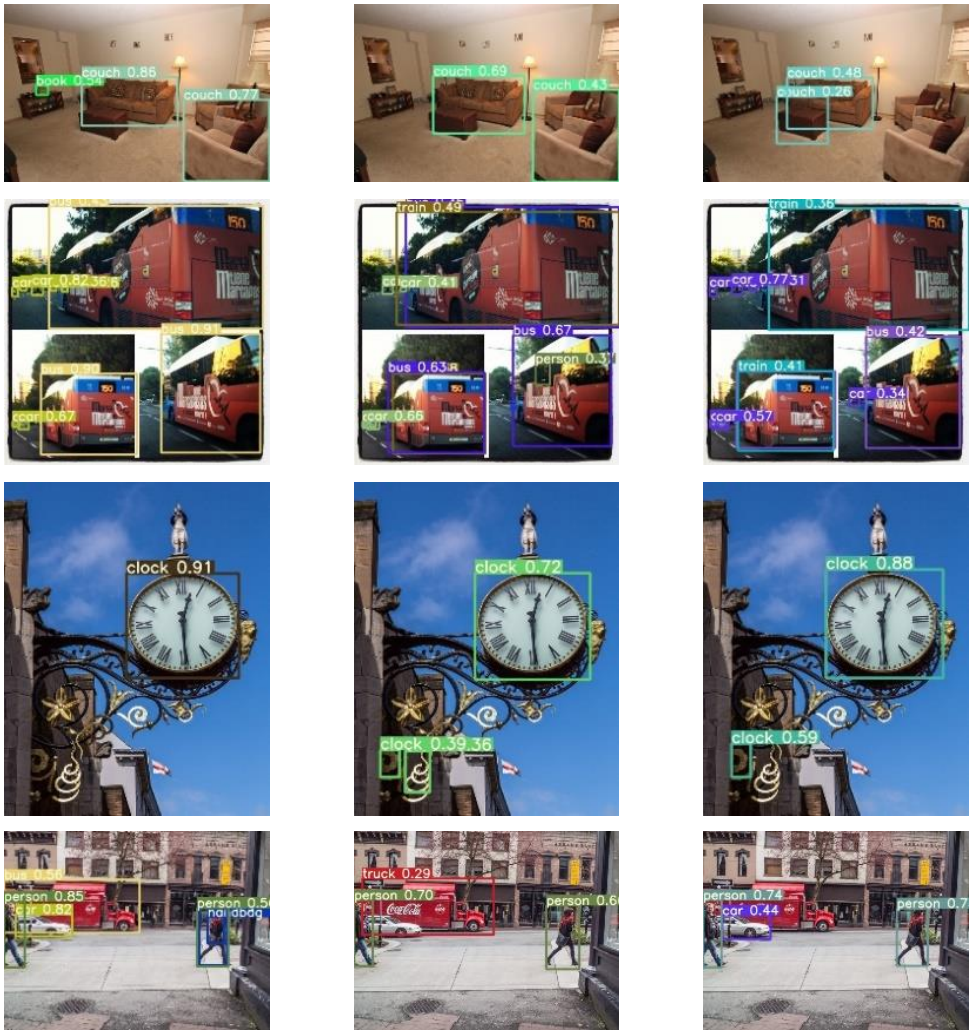


Figure 6. Training process of the YOLO-L network model on the Minicoco dataset.



(**a**) (**b**) (**c**)

Figure 7. Some visual comparisons were conducted on the MS COCO test dataset. (**a**) Improved YOLO-L network. (**b**) Detection results of the Mobilenetv3 network model in Table 4. (**c**) Representative Shufflenetv2 network model detection results in Table 4.

5. Conclusions

According to the concept of a lightweight network model structure, referring to the network model structure of PP-LCNet, in this study, the technique was integrated into the backbone of YOLOv5, thereby reducing the model size, parameters, and computation. Inspired by the network model structure of ShuffleNetv2, the stacking method of depth and point convolutions, which is mainly used in PP-LCNet, was improved, and the accuracy of the model was enhanced. Finally, CBAM was added to the neck network, and according to the conclusion drawn in PP-LCNet, the end convolution module in the neck network was replaced by the end depth-separable convolution module with the SE module in PP-LCNet, which ultimately enhanced the model's accuracy and significantly improved detection speed on low-performance devices. In future work, we will consider transplanting it to specific engineering equipment and use different datasets for training to obtain different lightweight detection models and apply them to different target detection tasks.

Author Contributions: Conceptualization, T.W. and S.S.; methodology, X.D.; software, X.D.; validation, X.D.; resources, T.W.; writing—original draft preparation, X.D.; writing—review and editing, S.S.; visualization, X.D.; supervision, X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the joint fund project of the National Natural Science Foundation of China (Grant number U20A20201), the Autonomous Project of State Key Laboratory of Robotics (Grant number 2022-Z02, 2022-Z19), the Liaoning Province Applied Basic Research Program Project (Grant number 2023JH2/101300141), and the National Science Foundation of China (Grant number 62103406).

Data Availability Statement: Not applicable.

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

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
2. Jin, H.; Wu, D.; Zhang, S.; Zou, X.; Jin, S.; Tao, D.; Liao, Q.; Xia, W. Design of a Quantization-Based DNN Delta Compression Framework for Model Snapshots and Federated Learning. *IEEE Trans. Parallel Distrib. Syst.* **2023**, *34*, 923–937. [CrossRef]
3. Hallaji, E.; Razavi-Far, R.; Saif, M. Federated and transfer learning: A survey on adversaries and defense mechanisms. *Adapt. Learn. Optim.* **2023**, *27*, 29–55.
4. Liu, W.; Quijano, K.; Crawford, M.M. YOLOv5-Tassel: Detecting Tassels in RGB UAV Imagery with Improved YOLOv5 Based on Transfer Learning. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 8085–8094. [CrossRef]
5. Li, S.; Sultonov, F.; Tursunboev, J.; Park, J.-H.; Yun, S.; Kang, J.-M. Ghostformer: A GhostNet-Based Two-Stage Transformer for Small Object Detection. *Sensors* **2022**, *22*, 6939. [CrossRef]
6. Jiao, L.; Zhang, F.; Liu, F.; Yang, S.; Li, L.; Feng, Z.; Qu, R. A Survey of Deep Learning-Based Object Detection. *IEEE Access* **2019**, *7*, 128837–128868. [CrossRef]
7. Nguyen, K.; Todorovic, S. iFS-RCNN: An Incremental Few-shot Instance Segmenter. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–20 June 2022; pp. 7000–7009.
8. Girshick, R. Fast R-CNN. In Proceedings of the International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; IEEE Computer Society: Washington, DC, USA, 2015.
9. Wang, T.; Zhang, X.; Yuan, L.; Feng, J. Few-Shot Adaptive Faster R-CNN. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 7166–7175.
10. Cao, G.; Xie, X.; Yang, W.; Liao, Q.; Shi, G.; Wu, J. Feature-fused SSD: Fast detection for small objects. In Proceedings of the SPIE 10615, Ninth International Conference on Graphic and Image Processing (ICGIP 2017), 106151E (10 April 2018), Qingdao, China, 14–16 October 2017. [CrossRef]
11. Peng, Y.; Zhao, X.; Tao, H.; Liu, X.; Li, T. Gesture recognition based on deep learning in complex background. *Robot* **2019**, *41*, 534–542. [CrossRef]
12. Ruby, U.; Yendapalli, V. Binary cross entropy with deep learning technique for Image classification. *Int. J. Adv. Trends Comput. Sci. Eng.* **2020**, *9*, 5393–5397.
13. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
14. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934v1.
15. Li, H.; Wu, Z.; Zhu, C.; Xiong, C.; Socher, R.; Davis, L.S. Learning From Noisy Anchors for One-Stage Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020; pp. 10585–10594.
16. Cui, C.; Gao, T.; Wei, S.; Du, Y.; Guo, R.; Dong, S.; Lu, B.; Zhou, Y.; Lv, X.; Liu, Q.; et al. PP-LCNet: A Lightweight CPU Convolutional Neural Network. *arXiv* **2021**, arXiv:abs/2109.15099.
17. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
18. Fu, X.; Cao, X. Underwater image enhancement with global–local networks and compressed-histogram equalization. *Signal Process. Image Commun.* **2020**, *86*, 115892. [CrossRef]
19. Guo, W.; Li, W.; Li, Z.; Gong, W.; Cui, J.; Wang, X. A slimmer network with polymorphic and group attention modules for more efficient object detection in aerial images. *Remote Sens.* **2020**, *12*, 3750. [CrossRef]
20. Liang, T.; Glossner, J.; Wang, L.; Shi, S.; Zhang, X. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing* **2021**, *461*, 370–403. [CrossRef]
21. He, Y.; Zhang, X.; Sun, J. Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; Volume 2017, pp. 1398–1406.
22. Liu, Y.; Zhang, X.-Y.; Bian, J.-W.; Zhang, L.; Cheng, M.-M. SAMNet: Stereoscopically Attentive Multi-Scale Network for Lightweight Salient Object Detection. *IEEE Trans. Image Process.* **2021**, *30*, 3804–3814. [CrossRef]
23. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M. Searching for MobileNetV3. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.

24. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6848–6856.
25. Lan, H.; Meng, J.; Hundt, C.; Schmidt, B.; Deng, M.; Wang, X.; Liu, W.; Qiao, Y.; Feng, S. Feathercnn: Fast inference computation with tensorgemm on arm architectures. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *31*, 580–594. [CrossRef]
26. Liu, J.; Li, Q.; Cao, R.; Tang, W.; Qiu, G. Mininet: An extremely lightweight convolutional neural network for real-time unsupervised monocular depth estimation. *ISPRS J. Photogramm. Remote Sens.* **2020**, *166*, 255–267. [CrossRef]
27. Cui, B.; Dong, X.; Zhan, Q.; Peng, J.; Sun, W. Litedepthwisenet: A lightweight network for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5502915. [CrossRef]
28. Huang, Z.; Li, W.; Xia, X.; Wang, H.; Jie, F.; Tao, R. Lo-det: Lightweight oriented object detection in remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2021**, *60*, 5603515. [CrossRef]
29. Ke, X.; Lin, X.; Qin, L. Lightweight convolutional neural network-based pedestrian detection and re-identification in multiple scenarios. *Mach. Vis. Appl.* **2021**, *32*, 46. [CrossRef]
30. Rani, E. Littleyolo-spp: A delicate real-time vehicle detection algorithm. *Optik* **2021**, *225*, 165818.
31. Dai, W.; Li, D.; Tang, D.; Jiang, Q.; Wang, D.; Wang, H.; Peng, Y. Deep learning assisted vision inspection of resistance spot welds. *J. Manuf. Process.* **2021**, *62*, 262–274. [CrossRef]
32. Li, Y.; Li, M.; Qi, J.; Zhou, D.; Zou, Z.; Liu, K. Detection of typical obstacles in orchards based on deep convolutional neural network. *Comput. Electron. Agric.* **2021**, *181*, 105932. [CrossRef]
33. Lyu, B.; Yuan, H.; Lu, L.; Zhang, Y. Resource-constrained neural architecture search on edge devices. *IEEE Trans. Network Sci. Eng.* **2021**, *9*, 134–142. [CrossRef]
34. Zhao, Y.; Yin, Y.; Gui, G. Lightweight deep learning based intelligent edge surveillance techniques. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *6*, 1146–1154. [CrossRef]
35. Qin, H.; Shao, S.; Wang, T.; Yu, X.; Jiang, Y.; Cao, Z. Review of Autonomous Path Planning Algorithms for Mobile Robots. *Drones* **2023**, *7*, 211. [CrossRef]
36. Xia, X.; Meng, Z.; Han, X.; Li, H.; Tsukiji, T.; Xu, R.; Zheng, Z.; Ma, J. An automated driving systems data acquisition and analytics platform. *Transp. Res. Part C Emerg. Technol.* **2023**, *151*, 104120. [CrossRef]
37. Woo, J.; Baek, J.-H.; Jo, S.-H.; Kim, S.Y.; Jeong, J.-H. A Study on Object Detection Performance of YOLOv4 for Autonomous Driving of Tram. *Sensors* **2022**, *22*, 9026. [CrossRef]
38. Bai, J.; Li, S.; Huang, L.; Chen, H. Robust Detection and Tracking Method for Moving Object Based on Radar and Camera Data Fusion. *IEEE Sens. J.* **2021**, *21*, 10761–10774. [CrossRef]
39. Zhao, Y.; Cheng, J.; Zhou, W.; Zhang, C.; Pan, X. Infrared pedestrian detection with converted temperature map. In Proceedings of the 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Lanzhou, China, 18–21 November 2019; pp. 2025–2031.
40. Zulkhairi, M.A.; Mustafah, Y.M.; Abidin, Z.Z.; Zaki, H.F.M.; Rahman, H.A. Car detection using cascade classifier on embedded platform. In Proceedings of the 2019 7th International Conference on Mechatronics Engineering (ICOM), Putrajaya, Malaysia, 30–31 October 2019; pp. 1–3.
41. Zhu, X.; Lyu, S.; Wang, X.; Zhao, Q. TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Montreal, BC, Canada, 11–17 October 2021; pp. 2778–2788. [CrossRef]
42. Liu, R.; Ren, C.; Fu, M.; Chu, Z.; Guo, J. Platelet Detection Based on Improved YOLO_v3. *Cyborg Bionic Syst.* **2022**, *2022*, 9780569. [CrossRef]
43. Xue, M.; Chen, M.; Peng, D.; Guo, Y.; Chen, H. One Spatio-Temporal Sharpening Attention Mechanism for Light-Weight YOLO Models Based on Sharpening Spatial Attention. *Sensors* **2021**, *21*, 7949. [CrossRef] [PubMed]
44. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv* **2019**, arXiv:1905.11946.
45. Li, Z.; Wang, S.; Fan, R.; Cao, G.; Zhang, Y.; Guo, T. Teeth category classification via seven-layer deep convolutional neural network with max pooling and global average pooling. *Int. J. Imaging Syst. Technol.* **2019**, *29*, 577–583. [CrossRef]
46. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.