agriculture

MDPI

Article

A Lightweight Detection Method for Blueberry Fruit Maturity Based on an Improved YOLOv5 Algorithm

Feng Xiao, Haibin Wang *, Yueqin Xu and Zhen Shi

College of Mechanical and Electrical Engineering, Northeast Forestry University, Harbin 150040, China; xiaofeng@nefu.edu.cn (F.X.)
* Correspondence: whb_nefu@nefu.edu.cn

Abstract: In order to achieve accurate, fast, and robust recognition of blueberry fruit maturity stages for edge devices such as orchard inspection robots, this research proposes a lightweight detection method based on an improved YOLOv5 algorithm. In the improved YOLOv5 algorithm, the ShuffleNet module is used to achieve lightweight deep-convolutional neural networks. The Convolutional Block Attention Module (CBAM) is also used to enhance the feature fusion capability of lightweight deep-convolutional neural networks. The effectiveness of this method is evaluated using the blueberry fruit dataset. The experimental results demonstrate that this method can effectively detect blueberry fruits and recognize their maturity stages in orchard environments. The average recall (*R*) of the detection is 92.0%. The mean average precision (mAP) of the detection at a threshold of 0.5 is 91.5%. The average speed of the detection is 67.1 frames per second (fps). Compared to other detection algorithms, such as YOLOv5, SSD, and Faster R-CNN, this method has a smaller model size, smaller network parameters, lower memory usage, lower computation usage, and faster detection speed while maintaining high detection performance. It is more suitable for migration and deployment on edge devices. This research can serve as a reference for the development of fruit detection systems for intelligent orchard devices.

Keywords: blueberry fruit; deep learning; machine vision; object detection; YOLOv5

Citation: Xiao, F.; Wang, H.; Xu, Y.; Shi, Z. A Lightweight Detection Method for Blueberry Fruit Maturity Based on an Improved YOLOv5 Algorithm. *Agriculture* **2024**, *14*, 36. https://doi.org/10.3390/agriculture14010036

Academic Editors: Gniewko Niedbała, Sebastian Kujawa, Tomasz Wojciechowski and Magdalena Piekutowska

Received: 21 November 2023
Revised: 20 December 2023
Accepted: 20 December 2023
Published: 24 December 2023

1. Introduction

Blueberry fruits are rich in many nutrients, particularly vitamin C, vitamin K, and manganese. In addition, they are rich in fiber, anthocyanins, antioxidants, and other bioactive compounds. These ingredients provide blueberry fruits with a variety of health benefits, including the ability to enhance immune function and improve cardiovascular health. Figure 1 shows blueberry plants and their fruits. As people pay increasing attention to their health, the demand for blueberry fruits is also rising. Consequently, the commercial cultivation areas for blueberry fruits are expanding [1].

Blueberry fruit harvesting directly impacts the yield and income generated from blueberry cultivation. Blueberry fruits have a short ripening period, which typically occurs during the rainy season. The contradiction between the high demand for labor and the shortage of labor in the traditional blueberry production modes has become increasingly evident. The labor cost of blueberry harvesting has reached 30–50% of the total production cost [2]. To some extent, it has hindered the development of the blueberry industry [3] and agricultural production. Developing precision agriculture and smart agriculture technology is an important step in addressing the labor-intensive nature of the blueberry industry [4–6]. Accurately, quickly, and robustly detecting blueberry fruits and providing information on their maturity stages are essential requirements for efficient and timely harvesting. Currently, researchers have conducted extensive studies on fruit detection and recognition using deep learning (DL) technology [7–9]. Based on the YOLOv2 algorithm, Xiong J. et al. [10] detected green mangoes on the surface of tree crowns. They

achieved a precision of 96.1% and a recall of 89.0%. Based on the YOLOv3 algorithm, Zhang W. et al. [11] counted the number of citrus fruits in video sequences. They resolved the issue of double-counting fruit in overlapping situations. Based on the YOLOv4 algorithm, Gao F. et al. [12] detected and counted apples by tracking their stems. They achieved a mean average precision of 99.35% for fruits and trunks. Miao et al. [13] integrated classical image processing techniques with the YOLOv5 algorithm. The detection and recognition performance of the YOLOv5 algorithm has been significantly improved. In addition, Yu Y. et al. [14] detected strawberries using the Mask R-CNN algorithm, while Jia W. et al. [15] detected apples.

Figure 1. Blueberry plants and their fruits in an orchard.

In order to achieve accurate, fast, and robust recognition of blueberry fruit maturity stages for orchard inspection robots while also enhancing the deployment capability of the blueberry fruit detection algorithm on edge devices such as agricultural unmanned aerial vehicles (UAVs) and agricultural unmanned ground vehicles (UGVs), this research proposes a lightweight detection method based on an improved YOLOv5 algorithm. It uses 680 images of blueberry fruits, including 9935 target blueberry fruits, to evaluate the effectiveness of the improved YOLOv5 algorithm and compares its performance with other advanced fruit detection algorithms, such as YOLOv5, SSD, and Faster R-CNN.

Section 2 introduces the production of the blueberry fruit dataset, the YOLOv5 algorithm, and the improved YOLOv5 algorithm. Section 3 demonstrates the detection performance of the improved YOLOv5 algorithm. To investigate the impact of each module changed in the improved YOLOv5 algorithm, Section 3 also conducted a comprehensive ablation study. Section 4 concludes the paper.

2. Materials and Methods

2.1. Dataset Production

2.1.1. Data Acquisition

The blueberry fruit dataset, which serves as the signal source guiding deep convolutional neural networks for blueberry fruit detection, plays a significant role in determining the overall performance of algorithms. The blueberry fruit images in the dataset came from two sources. Some were collected from the published dataset in Reference [5]. The others were collected from an orchard by the camera of an iPhone 11 set to fully automatic mode. The orchard is located in the Jizhou District, Tianjin, China.

Considering the planting spacing of blueberry plants in the orchard, the photographers positioned themselves at a distance of 0.4–0.9 m from the blueberry plants. The distance between the camera and the blueberry plants is 0.2–0.5 m, and the camera is approximately 1.2 m above the ground. The three shooting distances of near, medium, and far correspond to 0.2–0.3 m, 0.3–0.4 m, and 0.4–0.5 m, respectively, from the camera to the blueberry plants. Examples of blueberry fruit images at the three shooting distances are shown in Figure 2.

(**a**) near distance (**b**) medium distance (**c**) far distance

Figure 2. Examples of blueberry fruit images at the three shooting distances.

The working environment of edge devices in orchards is also affected by external conditions, such as the presence of branches and leaves. As shown in Figure 3, the blueberry fruit images in the dataset can be classified into various types. These types include mild clustering, severe clustering, mild occlusion, severe occlusion, backlighting, and blurred background. In addition, the blueberry fruit images in the dataset include different time periods (such as morning and afternoon) and various weather conditions (such as sunny and cloudy).

(**a**) mild clustering (**b**) severe clustering

(**c**) mild occlusion (**d**) severe occlusion

(**e**) backlighting (**f**) blurred background

Figure 3. Examples of blueberry fruit images in different conditions.

2.1.2. Data Preprocessing

The ripeness of blueberry fruits is mainly judged based on their color. According to the sensory evaluation method and the expertise of blueberry cultivation experts, blueberry fruits typically go through three stages of ripening: fully ripe, semi-ripe, and immature. The color of immature blueberry fruits is similar to that of the branches and leaves of blueberry plants, while the color of fully ripe blueberry fruits is similar to that of the soil. LabelImg is a graphical image annotation tool. It was created by Tzutalin at National Taiwan University with the help of dozens of contributors. In this research, LabelImg is used to label the original blueberry fruit images according to the labeling format of the

Pascal VOC dataset and generate .xml-type labeling files. Fully ripe blueberry fruits are labeled as "blueberry", semi-ripe blueberry fruits are labeled as "blueberry-halfripe", and immature blueberry fruits are labeled as "blueberry-unripe". When labeling, the bounding box used is the smallest rectangle that completely encloses the entire blueberry fruit. This is done to minimize the inclusion of unnecessary pixels from the background.

Training deep convolutional neural networks requires a large amount of data [16]. Too little data can result in underfitting or overfitting of deep convolutional neural networks. Therefore, data augmentation needs to be performed on the original blueberry fruit images. As shown in Figure 4, this research employs various methods, including mirroring, rotation, scaling, adding noise, and adjusting brightness, to enhance the diversity of the blueberry fruit images that we have collected. The annotation files corresponding to each image are transformed simultaneously. The augmented blueberry fruit dataset contains 680 images and can improve the robustness of deep convolutional neural networks in detecting the ripeness of blueberry fruits in orchards.

(**a**) original (**b**) mirroring

(**c**) rotation (**d**) scaling

(**e**) adding noise (**f**) adjusting brightness

Figure 4. Data augmentation.

The blueberry fruit dataset is randomly divided into a training set and a validation set [17]. The data distribution is shown in Table 1. The training set contains 544 images of blueberry fruits, including 7895 target blueberry fruits. Among these, there are 4310 fully mature blueberry fruits, 655 semi-ripe blueberry fruits, and 2930 immature blueberry fruits. The validation set contains 136 images of blueberry fruits, including 2040 target blueberry fruits. Among these images, there are 1169 fully mature blueberry fruits, 172 semi-ripe blueberry fruits, and 699 immature blueberry fruits.

Table 1. Data distribution.

Datasets	Number of Blueberry Fruit Images	Number of Target Blueberry Fruits		
		Total	Types	Number
blueberry fruit dataset	680	9935	mature semi-ripe immature	5479 827 3629
training set	544	7895	mature semi-ripe immature	4310 655 2930
validation set	136	2040	mature semi-ripe immature	1169 172 699

2.2. The YOLOv5 Algorithm

The YOLOv5 algorithm is one of the algorithms in the YOLO series [18]. It is an improvement based on the YOLOv4 algorithm. The YOLOv5 algorithm consists of 10 detectors: YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x, and others. The main differences among them lie in the number of convolutional layers and optimal application scenarios. As the number of convolutional layers increases, the model size gradually increases, while the detection performance improves and the detection speed decreases. This research focuses on the YOLOv5s 7.0 model as the subject. As shown in Figure 5, the overall network structure of the YOLOv5s 7.0 model can be divided into four parts: the input layers, the backbone feature extraction networks, the neck enhancement feature extraction networks, and the output layers.

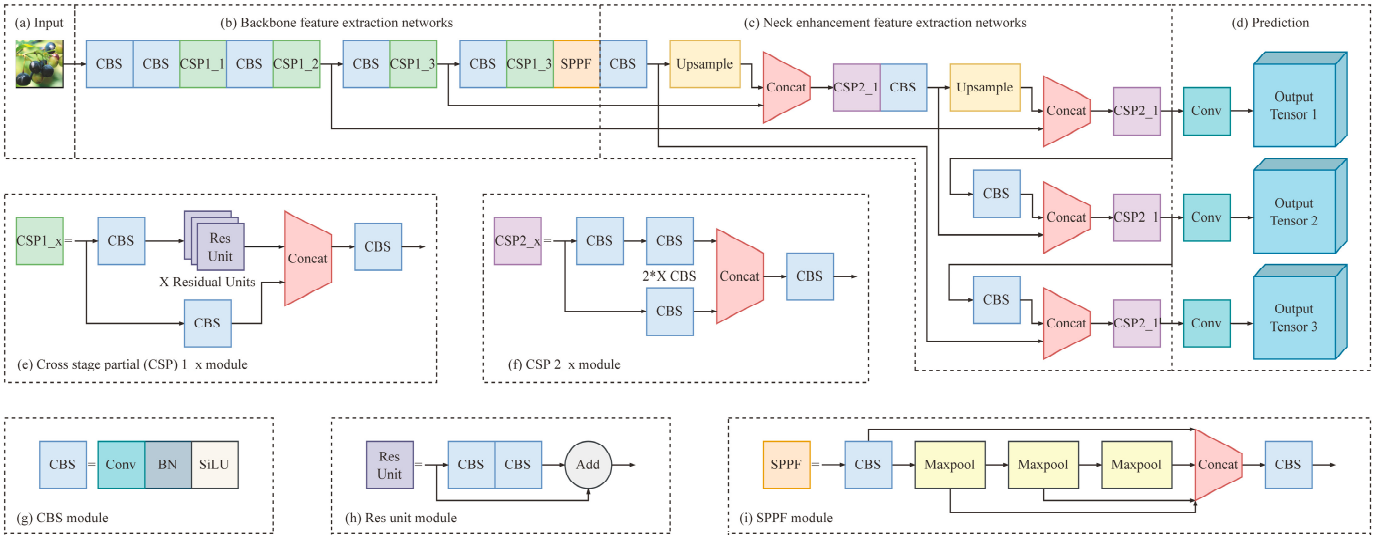


Figure 5. Structure of the YOLOv5 algorithm.

The input layers of the YOLOv5 model include Mosaic data enhancement, adaptive image scaling, and adaptive anchor box calculation [19]. The Mosaic data enhancement operation splices input images using random scaling, random cropping, and random arrangement [20]. It enhances the effectiveness of detecting and recognizing small targets. The adaptive image scaling operation adds minimal black borders to the original images while calculating the scaling ratio, scaled size, and padding value for the black borders. In training deep convolutional neural networks, the optimal anchor box is calculated using the adaptive anchor box calculation operation. Compared to the method of presetting the anchor box length and width, this calculation operation minimizes the difference between the anchor box and the ground truth box. Additionally, through reverse updating, the

parameters of deep convolutional neural networks are optimized, resulting in improved performance for target detection and recognition.

In the backbone feature extraction networks of the YOLOv5 model, the CBS module is responsible for extracting features from images and organizing feature maps. It involves enhancing dimensionality, reducing dimensionality, downsampling, and normalization for feature maps. As shown in Figure 5g, a CBS module consists of a Conv2d function, a BatchNorm2d function, and a SiLU activation function [21,22]. The padding of the Conv2d function in the CBS module is automatically calculated. The stride of the Conv2d function in the CBS module is set to 2, and the kernel size is set to 3. Therefore, during the process of downsampling the feature map to extract the target features, the CBS module reduces the width and height of the feature map by half each time. The BatchNorm2d function is a layer for batch normalization, which normalizes the input data in each batch. The expressions for the SiLU activation function and its derivative are as follows:

$$SiLU(x) = f(x) = x \cdot sigmoid(x) = x/(1 + e^{-x}) \quad (1)$$

$$SiLU'(x) = f'(x) = x \cdot f(x) \cdot (1 - f(x)) + f(x) \quad (2)$$

where x represents the variable, SiLU(x) represents the SiLU activation function, and SiLU$'$(x) represents the derivative of the SiLU activation function. The graph of the SiLU activation function and its derivative is shown in Figure 6. The SiLU activation function has the characteristics of being unbounded, smooth, and non-monotonic. It increases the nonlinearity of the input data. To be specific, the SiLU activation function remains continuously differentiable as x approaches 0, which helps to avoid the vanishing gradient problem of deep convolutional neural networks. The SiLU activation function has a weak negative response at $x < 0$, which promotes the generalization ability of deep convolutional neural networks.

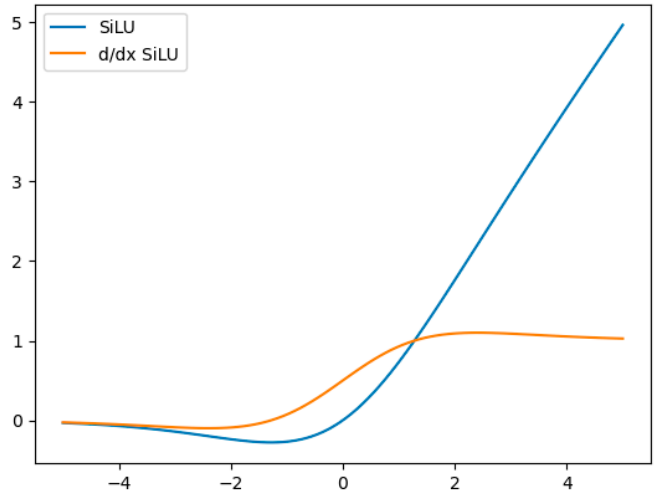


Figure 6. Graph of the SiLU activation function and its derivative.

As convolution operations progress, the deep convolutional neural networks extract increasingly complex feature information. In the shallow layers of the deep convolutional neural networks, the networks extract relatively simple feature information, such as color, shape, and texture. These features are visible in graphics, so they are referred to as graphical features. Next, the deeper convolutional neural networks fuse these graphical features and expand their dimensions to create new features. The new features are referred to as semantic features. Graphical features are simplistic and lack semantic depth, while semantic features provide more comprehensive information but may overlook basic visual elements.

In the neck enhancement feature extraction networks of the YOLOv5 model, the structure of the Feature Pyramid Network (FPN) and Path Aggregation Network (PANet) is utilized to combine shallow graphical features with deep semantic features [23], as depicted

in Figure 7. Specifically, the FPN fuses low-level features by upsampling them to the top-level features and makes predictions on each fused feature layer. This approach combines the advantages of low-level and high-level features, effectively improving the performance of detecting and recognizing small targets. However, while the FPN effectively transmits semantic features from top to bottom, it does not transmit positional information. Therefore, the PANet is introduced. The PANet includes a bottom-up path augmentation structure based on the FPN. This structure utilizes shallow features in deep convolutional neural networks. The top feature maps can benefit from the abundant positional information provided by the lower layers. This improves the performance of detecting medium- and large-sized targets. The structure of the FPN and PANet effectively enhances the detection and recognition capabilities of deep convolutional neural networks for fruits of various types and sizes.

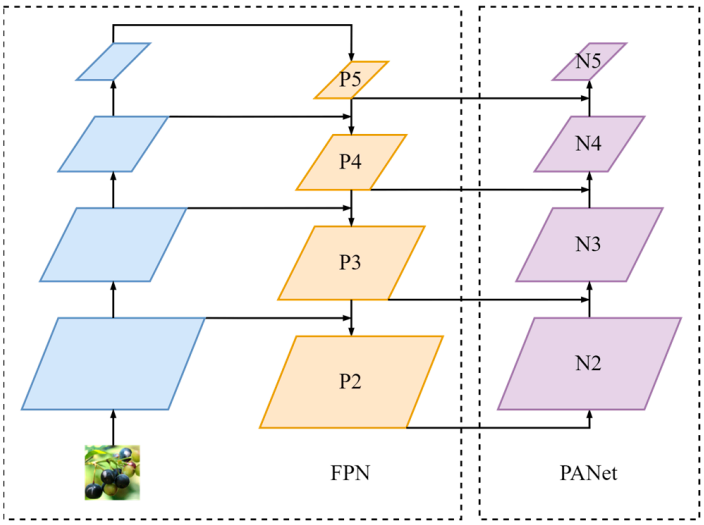


Figure 7. Structures of the FPN and PANet. (P2, P3, P4, and P5 indicate the feature layers generated by the FPN. N2, N3, N4, and N5 represent the feature layers generated by the PANet.)

The loss functions of the output layers of the YOLOv5 model consist of three components. The Generalized Intersection over Union (GIoU) loss function is used to calculate the loss for boundary regression [24]. The YOLOv5 model performs weighted Non-Maximum Suppression (NMS) on GIOU_Loss to achieve the efficient selection of the optimal bounding box. The Binary Cross Entropy with Logits (BCEWithLogits) loss function is used to calculate the loss for confidence prediction, while the Binary Cross Entropy loss (BCELoss) function is used to calculate the loss for class prediction.

2.3. The Improved YOLOv5 Algorithm

2.3.1. The ShuffleNet Module

In recent years, there have been some proposals for lightweight structures of deep convolutional neural networks, such as MobileNet and ShuffleNet. The ShuffleNetV1 is a lightweight deep-convolutional neural network proposed by MEGVII [25]. The modules of the ShuffleNetV1 are shown in Figure 8. In order to achieve a good balance between detection accuracy and speed, the ShuffleNetV1 utilizes the group convolution (GConv) operation and the channel shuffle operation.

Standard convolution is a method of densely connecting channels. This means that the feature information from each channel in the feature map of each layer is outputted to each channel of the feature map in the next layer through a convolution operation. The process is shown in Figure 9a. The parameters P for standard convolution are as follows:

$$P = D \times D \times M \times Z \quad (3)$$

where D represents the size of the convolution kernel, and M represents the number of input feature channels. Z represents the number of output feature channels, which is also equal to the number of convolution kernels.

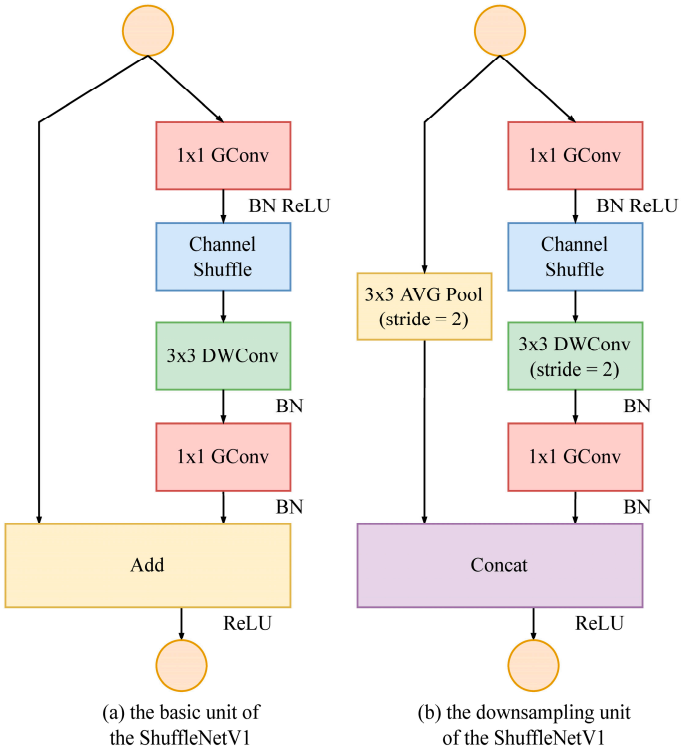
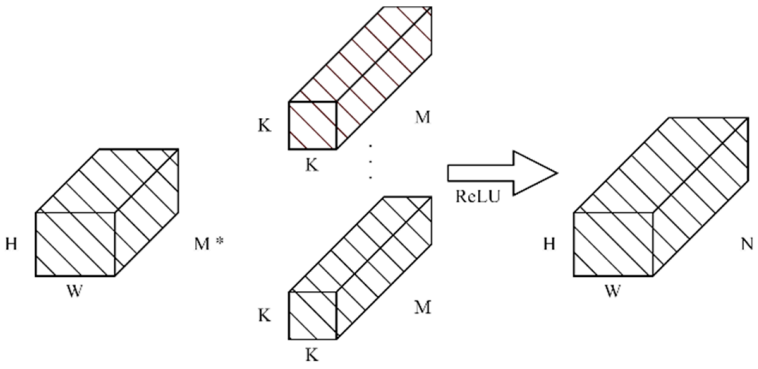
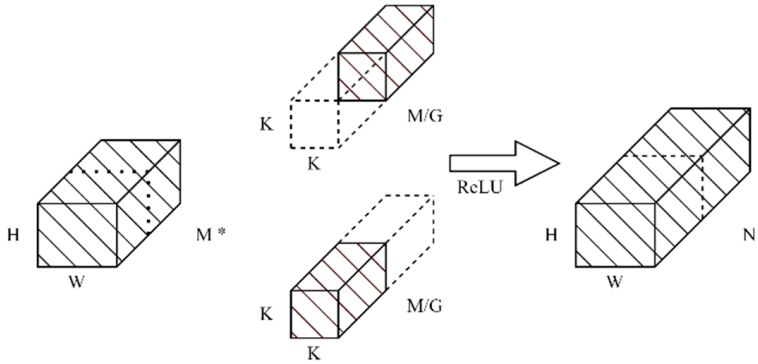


Figure 8. Modules of the ShuffleNetV1.



(**a**) standard convolution



(**b**) group convolution

Figure 9. Standard convolution and group convolution.

Group convolution is a method used to connect sparse convolutions. In this method, the input feature channel number is divided into G groups. Each group has M/G feature channels for the convolution kernel. After the convolution operation, each group produces a feature map with an output feature channel number of N. The process is illustrated in Figure 9b. The parameters P_{GC} for group convolution are as follows:

$$P_{GC} = D \times D \times M/G \times Z \tag{4}$$

Comparing Equations (3) and (4), the parameters and calculations for standard convolution are G of the parameters and calculations for group convolution. The group convolution operation, however, has limitations in terms of its ability to exchange information among groups. As shown in Figure 10, the channel shuffle operation reorganizes the feature information of various groups in the output layers to improve communication throughout the feature information of each group. This method enhances the learning capacity of feature information across groups and reduces the computational load of deep convolutional neural networks.

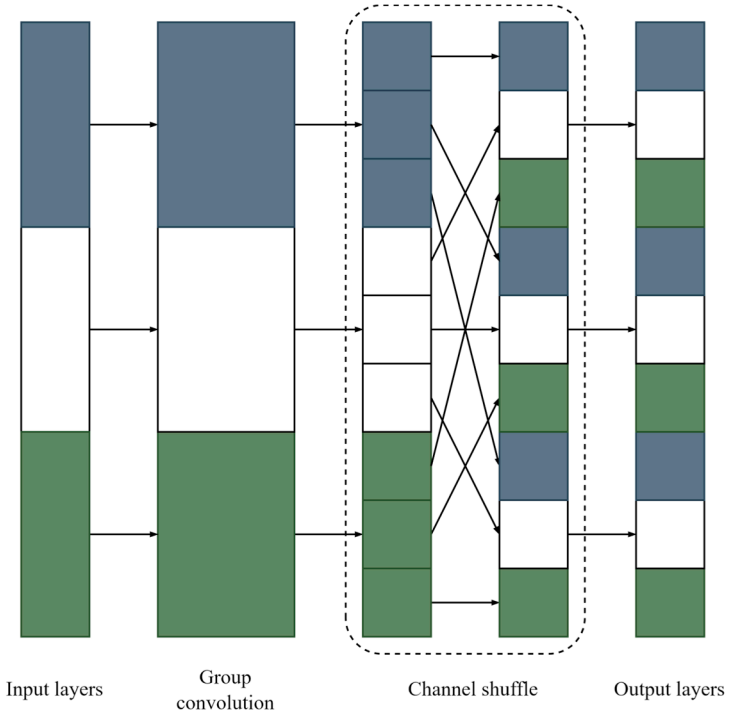


Figure 10. Channel shuffle schematic.

Ma et al. [26] designed the ShuffleNetV2 based on the ShuffleNetV1. As shown in Figure 11, the ShuffleNetV2 primarily consists of the basic unit and the downsampling unit. As shown in Figure 11a, the input feature channels in the basic unit of the ShuffleNetV2 are divided into two branches through the channel split operation. Each branch has an equal number of feature channels. The left branch performs identity mapping [27]. The right branch undergoes two 1×1 ordinary convolutions and one 3×3 depthwise convolution (DWConv) while maintaining an equal number of input and output channels. The left and right branches are merged through the channel concatenation operation, and the channel shuffle operation is performed to ensure the integration of feature information from both branches. The downsampling unit of the ShuffleNetV2, shown in Figure 11b, directly inputs the feature map into two branches. Each branch performs the 1×1 ordinary convolution and 3×3 DWConv with a stride of 2. After merging the two branches using the channel concatenation operation, the number of output channels is doubled. The merged feature map undergoes the channel shuffle operation. Unlike the basic unit, the downsampling

unit directly increases the number of feature channels in the deep convolutional neural networks and expands their width without using the channel split operation. This approach avoids increasing the computational burden of the deep convolutional neural networks while further enhancing their feature extraction capabilities.

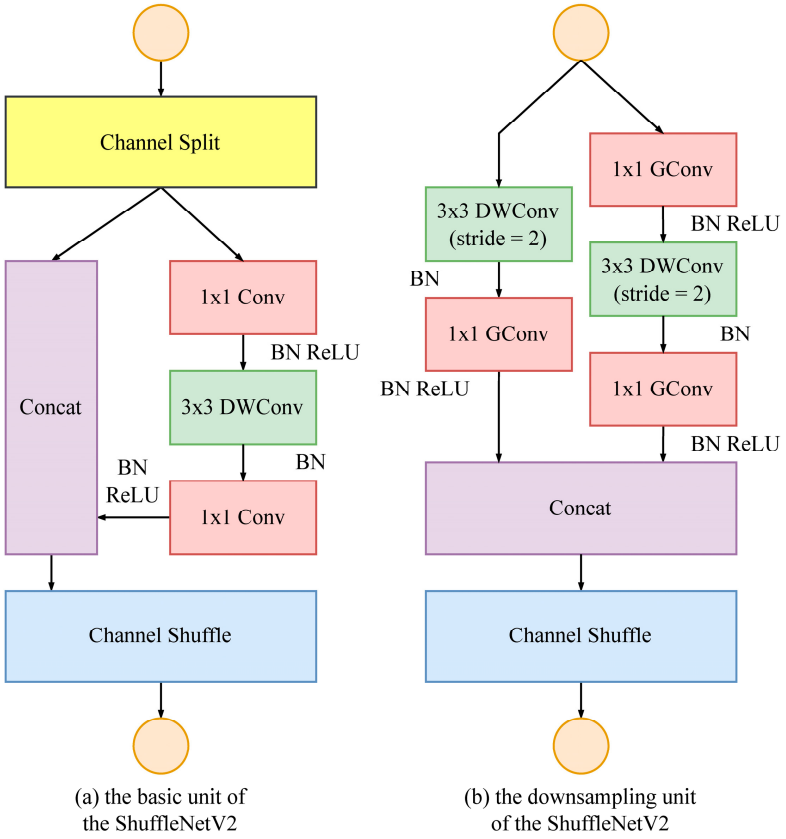


Figure 11. Modules of the ShuffleNetV2.

Under the same complexity, the ShuffleNetV2 outperforms the ShuffleNetV1, MobileNetV1, MobileNetV2, and other lightweight deep convolutional neural networks. The improved YOLOv5 algorithm was built upon the ShuffleNetV2, utilizing the SFB1_X and SFB2_X architectures. In the improved YOLOv5 algorithm, the backbone networks of the ShuffleNetV2 have replaced the 1024 convolution operation and the 5×5 pooling operation with the global average pooling operation. This change aims to improve the speed of detection and recognition by reducing the memory usage of the deep convolutional neural networks.

2.3.2. The CABM Module

Attention mechanisms originated from the study of human vision [28]. In computer vision (CV), attention mechanisms are used to process visual information. Traditional methods, such as the local feature extraction approach and the sliding window approach, can be considered as types of attention mechanisms. In DL, attention mechanisms are typically implemented as separate attention modules. Attention modules allow deep neural networks to assign different weights to different parts of the inputs, enabling them to focus on relevant units and suppress irrelevant units during the process of feature extraction. The Convolutional Block Attention Module (CBAM) [29], the Efficient Channel Attention Module (ECA) [30], and the Squeeze and Excitation Network (SENet) [31] are common attention mechanisms. The structure of the CBAM module is shown in Figure 12.

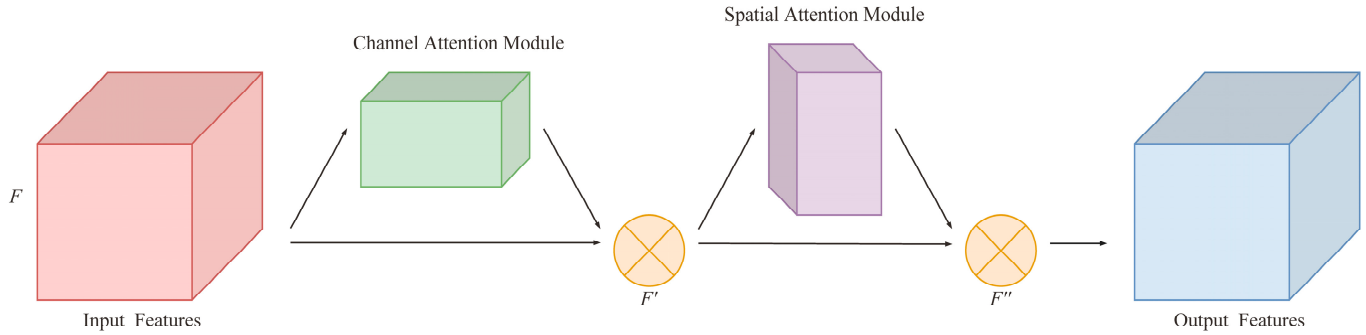


Figure 12. Structure of the CBAM module.

Compared to traditional attention mechanisms that solely focus on channels or spatial dimensions, the CBAM module consists of two parts: the Channel Attention Module (CAM), which focuses on channel information, and the Spatial Attention Module (SAM), which focuses on location information. The CBAM module combines channel information and spatial information to enable the deep convolutional neural networks to focus on important features and suppress interference from less significant ones. From our perspective, this will help address the issue of color similarity between immature blueberry fruits and their backgrounds, as well as occlusions caused by adjacent blueberry fruits and leaves. Therefore, this research used the CBAM module to enhance the performance of blueberry fruit detection and recognition. The overall attention process of the CBAM module used in this research is described by Equations (5) and (6):

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

$$F'' = M_s(F') \otimes F' \tag{6}$$

where F represents the original input feature map, F' represents the adjusted feature map using the CAM, F'' represents the final feature map using the SAM, M_c represents the weight matrix after channel compression, M_s represents the weight matrix after spatial compression, and $\otimes$ represents the element-wise multiplication of matrices.

2.3.3. The Improved YOLOv5 Algorithm

The overall network structure of the improved YOLOv5 algorithm is shown in Figure 13. It mainly consists of four parts: the input layers, the backbone feature extraction networks, the neck enhancement feature extraction networks, and the output layers. The input layers of the improved YOLOv5 algorithm accept blueberry fruit images with the size of 608 × 608 pixels. These images are then passed through the ShuffleNet modules in the backbone feature extraction networks of the improved YOLOv5 algorithm for blueberry fruit feature extraction. Subsequently, the blueberry fruit feature maps are sent to the neck enhanced feature extraction networks of the improved YOLOv5 algorithm for blueberry fruit feature fusion. Finally, the output layers of the improved YOLOv5 algorithm produce three prediction anchor boxes of different scales.

Compared to the YOLOv5 model:

(1) First, because the SPPF module needs to perform pooling operations at multiple scales and splice the results, it takes up more memory space. This limits the application of network models to resource-constrained devices. In order to achieve lightweight deep convolutional neural networks, the improved YOLOv5 algorithm removes the SPPF module from the backbone feature extraction networks of the YOLOv5 algorithm.

(2) Second, the CSP Bottleneck module utilizes the multi-channel separated convolution operation. Frequently using the CSP Bottleneck module can consume a significant amount of cache space and decrease the execution speed of deep convolutional neural networks. The ShuffleNet modules with Shuffle channels are used to replace the

CSPDarknet-53 modules in the backbone feature extraction networks of the YOLOv5 algorithm for blueberry fruit feature extraction.

(3) Finally, the CBAM modules are integrated into the neck enhancement feature extraction networks of the YOLOv5 algorithm to enhance the feature fusion capability of deep convolutional neural networks. This enables the efficient extraction of important features and the suppression of irrelevant ones.

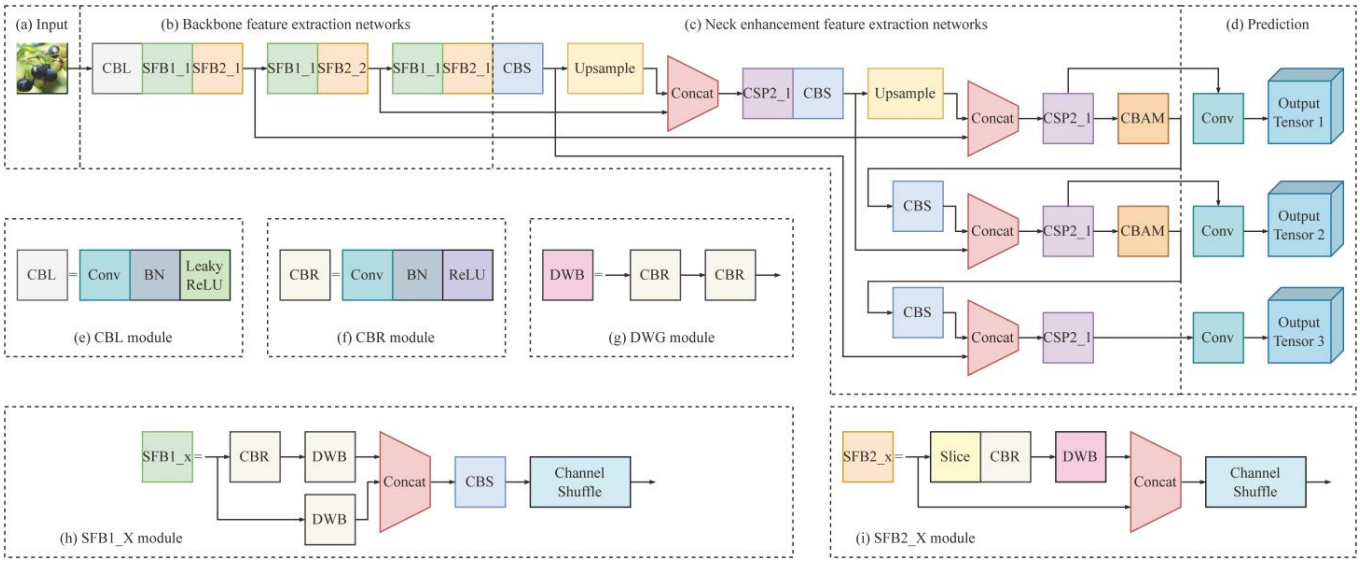


Figure 13. Structure of the improved YOLOv5 algorithm.

3. Results and Discussion

3.1. Experimental Platforms

The hardware platform used for the experiments is as follows: the CPU was a 13th Gen Intel® Core™ i5-13600K with 14 cores and 20 threads. It has a base frequency of 3.50 GHz and a maximum boost frequency of 5.10 GHz. The GPU was an NVIDIA GeForce RTX 3080 with 12 GB of memory and 8960 CUDA cores, which enable the accelerated training of deep convolutional neural networks. The memory consisted of two Hynix DDR5 5600 MHz 16 GB DIMMs (Dual In-Line Memory Modules). The hard disk was a Samsung SSD980 PRO with a capacity of 1 TB. The motherboard was an MSI PRO Z790-A WIFI DDR5.

The software configuration for the experiments is as follows: the operating system is Windows 11. The programming language is Python 3.8. The integrated development environment is PyCharm 2020.3.5. The DL framework is PyTorch 1.11.0. The parallel computer framework is CUDA 11.4.0, and the DL acceleration library is cuDNN 8.2.2.

In addition, this research sets the batch size to 32. The process of going through all the data in the blueberry fruit dataset once is referred to as one epoch, and the improved YOLOv5 algorithm runs for 1600 epochs. The values of the conf-thres parameter and iou-thres parameter are both set to 0.5.

3.2. Evaluation Metrics

To compare the performance of different fruit detection and recognition methods, the evaluation metrics of Precision (*P*), Recall (*R*), Average Precision (*AP*), and mean Average Precision (mAP) are used. The calculation formulas are shown in Equations (7)–(10). *P* represents the probability that the positive sample is accurately identified as the positive example by the classifier. *R* represents the classifier's ability to correctly identify all positive samples. The area enclosed by the *P*-*R* curve is represented by *AP*. The *P*-*R* curve is formed with the *R* as the independent variable and the *P* as the dependent variable. *mAP* is the average *AP* value across multiple categories [32]. It measures the classifier's ability to effectively detect and recognize all classes.

$$P = \frac{TP}{TP + FP} \tag{7}$$

$$R = \frac{TP}{TP + FN} \tag{8}$$

$$AP = \int_0^1 P(R)dR \tag{9}$$

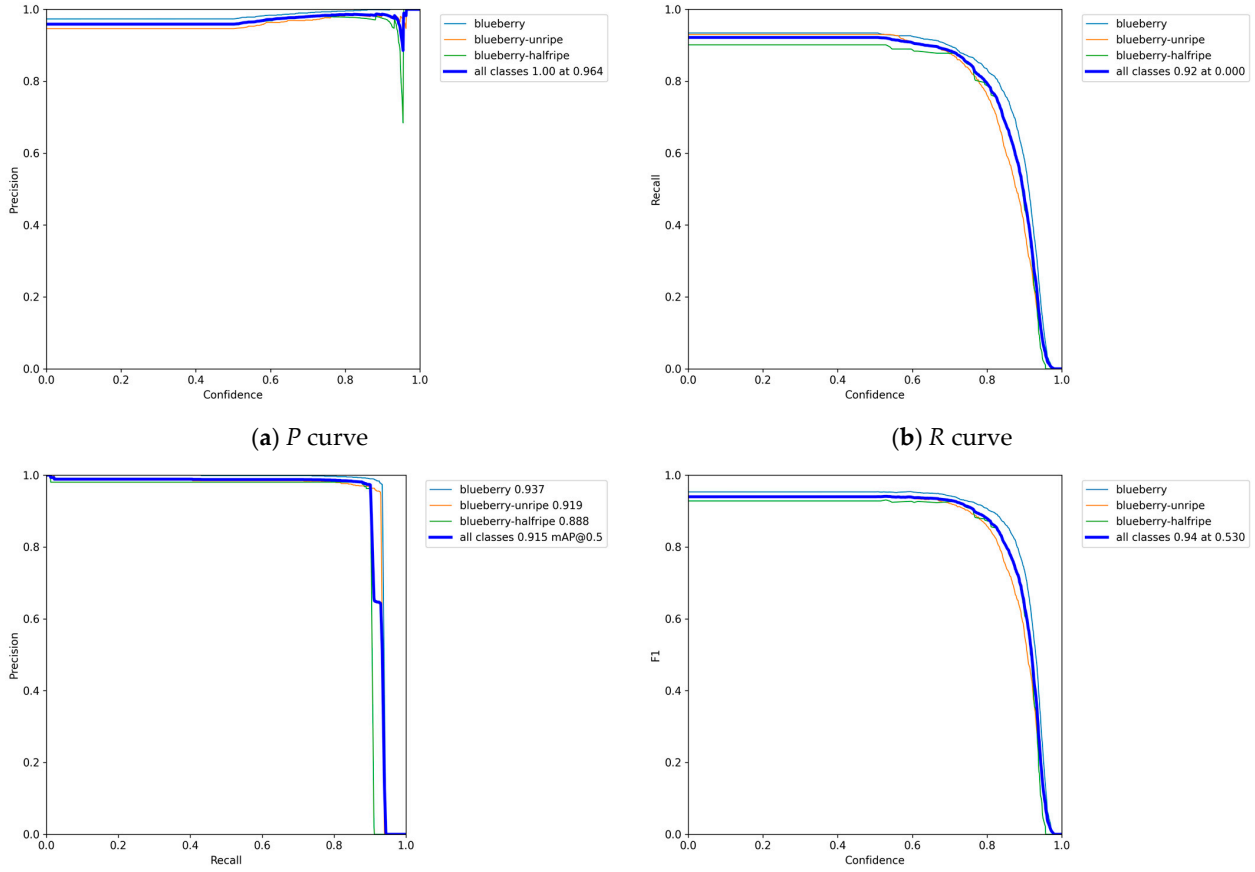
$$mAP = \frac{1}{N} \cdot \sum_{i=1}^{n} (AP_i) \tag{10}$$

where TP represents the number of correctly detected blueberry fruits, FP represents the number of erroneously detected blueberry fruits, FN represents the number of missed blueberry fruits, and N represents the number of categories for object detection.

In addition, the complexity, computational efficiency, and real-time performance of the improved YOLOv5 algorithm are evaluated by considering factors such as model size, network parameters, FLOPs, and detection speed.

3.3. Experimental Results

Experiments were conducted to test the detection effect of blueberry fruit ripeness. The *P* curve, *R* curve, *P*-*R* curve, and F1 curve of the improved YOLOv5 algorithm are shown in Figure 14. For the category of fully mature blueberry fruits with abundant samples, the mAP@0.5 are the highest, and there are few false detections and missed detections. The category with the second-highest mAP@0.5 is the immature blueberry fruit category, while the semi-ripe blueberry fruit category has the lowest mAP@0.5.

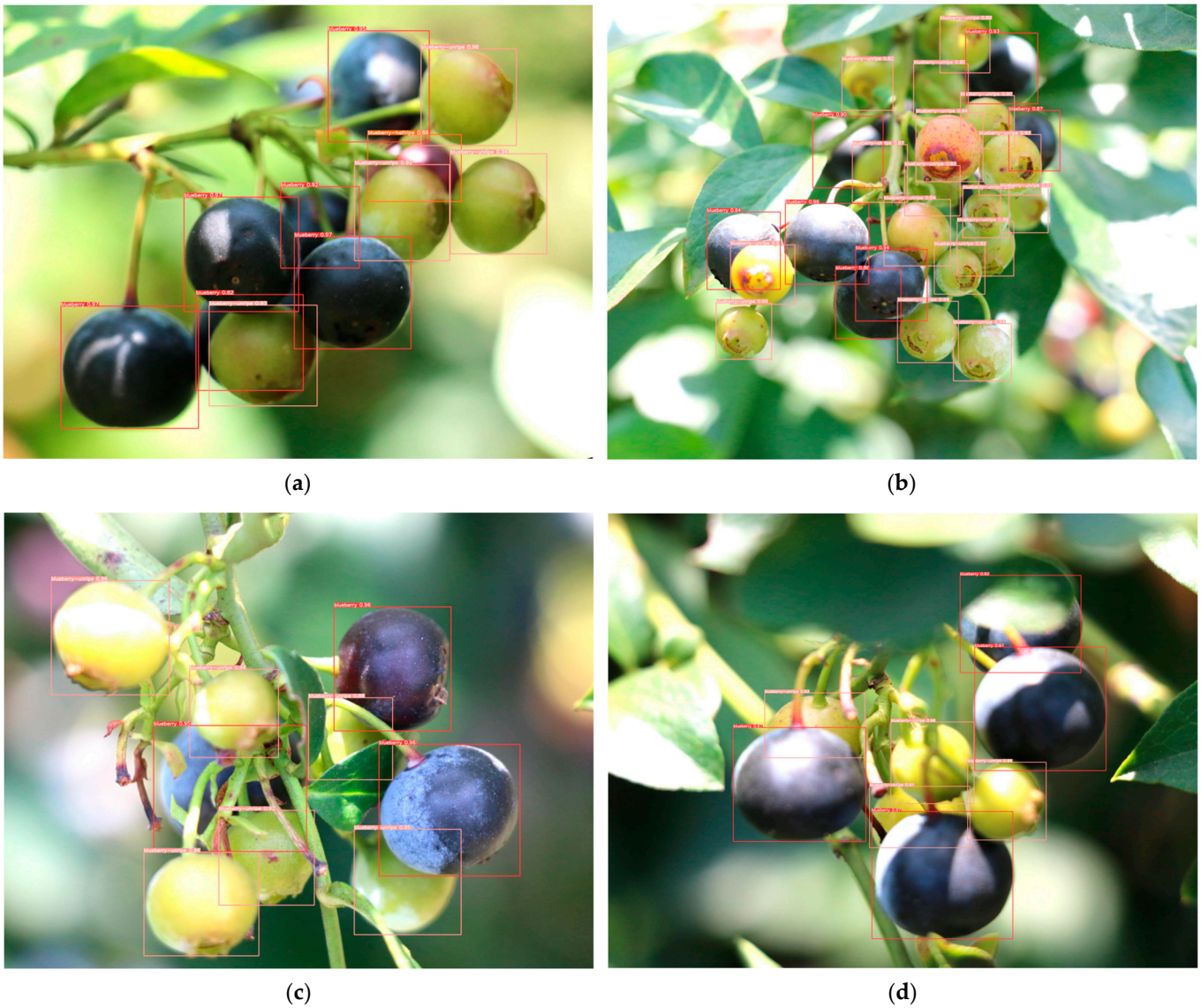


(**a**) *P* curve (**b**) *R* curve

(**c**) *P*-*R* curve (**d**) F1 curve

Figure 14. Experiment results of the improved YOLOv5 algorithm.

The examples of the detection effects of the improved YOLOv5 algorithm are shown in Figure 15. As shown in Figure 15a, the improved YOLOv5 algorithm demonstrates excellent detection results for blueberry fruits in close-range and medium-range images. It effectually detects blueberry fruits and recognizes their ripeness. As shown in Figure 15b, for blueberry fruits that are located at a greater distance, there is a higher likelihood of both false detection and missed detection. As shown in Figure 15c,d, the improved YOLOv5 algorithm can also effectually detect blueberry fruits, even in situations involving mild clustering, mild occlusion, backlighting, and blurred background. However, as shown in Figure 15b, the improved YOLOv5 algorithm may face challenges in effectually detecting blueberry fruits in scenarios with severe clustering and severe occlusion. This is because these instances do not provide enough distinctive feature information for classification.



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

(**c**) (**d**)

Figure 15. Examples of the detection effects of the improved YOLOv5 algorithm.

Generally speaking, the improved YOLOv5 algorithm achieves a *P* of 96.3%, a *R* of 92%, and a *mAP* of 91.5% at a threshold of 0.5. The average detection speed of the improved YOLOv5 algorithm is 67.1 fps with a batch size of 1 on the NVIDIA GeForce RTX 3080. The improved YOLOv5 algorithm has a 5.65 MB model size, 2.85 M network parameters, and 5.6 G FLOPs. It is suitable for migration and deployment on edge devices such as agricultural UAVs and agricultural UGVs.

3.4. Performance Comparison

This research also trains the YOLOv5, SSD, and Faster R-CNN algorithms using the blueberry fruit dataset. Their performance is then compared to the improved YOLOv5 algorithm. The performance of various blueberry fruit detection algorithms is presented in Table 2.

Table 2. Performance comparison of the various blueberry fruit detection algorithms.

Metrics/Models		YOLOv5	YOLOv5-Ours	SSD-vgg	Faster R-CNN-vgg
P (%)	mature	98.7	97.8	96.0	93.1
	semi-ripe	95.5	96.3	92.7	87.1
	immature	97.0	94.9	96.2	85.6
	mean value	97.1	96.3	95.0	88.6
R (%)	mature	93.5	92.9	96.0	95.8
	semi-ripe	91.3	90.1	89.0	90.1
	immature	93.4	93.0	93.9	93.0
	mean value	92.7	92.0	93.0	93.0
mAP@0.5 (%)	mature	95.1	93.7	95.9	95.6
	semi-ripe	91.0	88.8	88.0	89.1
	immature	93.5	91.9	92.5	91.0
	mean value	93.2	91.5	92.1	91.9
Model size (MB)		13.6	5.65	91.6	521.0
Parameter (M)		7.02	2.85	23.6	136.7
FLOPs (G)		15.8	5.6	246.6	376.5
Speed (fps)		66.2	67.1	44.4	17.0

As shown in Table 2, when compared to the YOLOv5 algorithm, the improved YOLOv5 algorithm exhibits a 0.8% decrease in precision, a 0.7% decrease in recall, and a 1.7% decrease in *mAP@0.5*. However, the model size, network parameter, and FLOPs of the improved YOLOv5 algorithm decrease by 7.95 MB, 4.17 M, and 10.2 G, respectively. The average detection speed of the improved YOLOv5 algorithm increases 0.9 fps.

Compared to the SSD-vgg, the improved YOLOv5 algorithm exhibits a 1.0% decrease in recall and a 0.6% decrease in *mAP@0.5*. However, the model size, network parameter, and FLOPs of the improved YOLOv5 algorithm decrease by 85.95 MB, 20.75 M, and 241.0 G, respectively. The precision and average detection speed of the improved YOLOv5 algorithm increase by 1.3% and 22.7 fps, respectively.

Compared to the Faster R-CNN-vgg, the improved YOLOv5 algorithm exhibits a 1.0% decrease in recall and a 0.4% decrease in *mAP@0.5*. However, the model size, network parameter, and FLOPs of the improved YOLOv5 algorithm decrease by 515.35 MB, 133.85 M, and 370.9 G, respectively. The precision and average detection speed of the improved YOLOv5 algorithm increase by 7.7% and 50.1 fps, respectively.

Generally speaking, when compared to the YOLOv5, SSD, and Faster R-CNN, the improved YOLOv5 algorithm has a smaller model size, smaller network parameters, lower memory usage, lower computation usage, and faster detection speed while maintaining high detection performance. It is more suitable for migration and deployment on edge devices.

For the purpose of testing the impact of each module changed in the improved YOLOv5 algorithm, this research conducted a comprehensive ablation study. The YOLOv5-ShuffleNet algorithm only replaced the CSPDarknet-53 module in the backbone feature extraction networks of the YOLOv5 algorithm with the ShuffleNetv2 module. The YOLOv5-CBAM algorithm only integrated the CBAM module into the neck enhancement feature extraction networks of the YOLOv5 algorithm. Table 3 presents the performance comparison among the YOLOv5, YOLOv5-ShuffleNet, YOLOv5-CBAM, and YOLOv5-ShuffleNet-CBAM algorithms.

Table 3. Results of the ablation study.

Metrics/Models		YOLOv5	YOLOv5-ShuffleNet	YOLOv5-CBAM	YOLOv5-ShuffleNet-CBAM
P (%)	mature	98.7	97.8	98.8	97.8
	semi-ripe	95.5	94.5	97.5	96.3
	immature	97.0	95.9	97.1	94.9
	mean value	97.1	96.1	97.8	96.3
R (%)	mature	93.5	90.8	96.1	92.9
	semi-ripe	91.3	89.5	90.6	90.1
	immature	93.4	88.3	95.1	93.0
	mean value	92.7	89.5	93.9	92.0
mAP@0.5 (%)	mature	95.1	91.6	96.5	93.7
	semi-ripe	91.0	88.8	94.0	88.8
	immature	93.5	87.2	90.4	91.9
	mean value	93.2	89.2	93.6	91.5
Model size (MB)		13.6	2.8	13.6	5.65
Parameter (M)		7.02	2.84	7.02	2.85
FLOPs (G)		15.8	5.5	15.8	5.6
Speed (fps)		66.2	77.0	57.1	67.1

As shown in Table 3, when compared to the YOLOv5 algorithm, the YOLOv5-Shuffle algorithm exhibits a reduction in model size, network parameters, and FLOPs by 10.8 MB, 4.18 M, and 10.3 G, respectively. The detection speed increases by 11.2 fps. When compared to the YOLOv5-CBAM algorithm, the YOLOv5-Shuffle-CBAM algorithm exhibits a reduction in model size, network parameters, and FLOPs by 7.95 MB, 4.17 M, and 10.2 G, respectively. The detection speed increases by 10 fps. The experimental results demonstrate that the ShuffleNet module can effectively achieve lightweight deep convolutional neural networks.

When compared to the Yolov5 algorithm, the YOLOv5-CBAM algorithm exhibits an increase of 0.7% in precision, a 1.2% increase in recall, and a 0.4% increase in *mAP@0.5*. When compared to the YOLOv5-ShuffleNet algorithm, the YOLOv5-ShuffleNet-CBAM algorithm demonstrates an improvement of 0.2% in precision, a 2.5% improvement in recall, and a 2.3% improvement in *mAP@0.5*. The experimental results demonstrate that the CBAM module can effectively enhance the feature extraction capability of deep convolutional neural networks.

4. Conclusions

Because most fruits mature in batches during their growth process, assessing fruit maturity is an important step in intelligent orchard management. Effective detection and statistics of fruit maturity are beneficial for planning fruit harvests and estimating fruit yields.

(1) This research proposes a lightweight detection method based on an improved YOLOv5 algorithm. First, in order to achieve lightweight deep convolutional neural networks, the improved YOLOv5 algorithm removes the SPPF module from the backbone feature extraction networks of the YOLOv5 algorithm. The ShuffleNet modules with Shuffle channels are used to replace the CSPDarknet-53 modules in the backbone feature extraction networks of the YOLOv5 algorithm for blueberry fruit feature extraction. Second, the CBAM modules are integrated into the neck enhancement feature extraction networks of the YOLOv5 algorithm to enhance the feature fusion capability of lightweight deep convolutional neural networks.

(2) The experimental results demonstrate that the improved YOLOv5 algorithm can effectively utilize RGB images to detect blueberry fruits and recognize their ripeness. The improved YOLOv5 algorithm achieves a *P* of 96.3%, an *R* of 92%, and a *mAP* of 91.5% at a threshold of 0.5. The average detection speed of the improved YOLOv5 algorithm is 67.1 fps with a batch size of 1 on the NVIDIA GeForce RTX 3080. The

improved YOLOv5 algorithm has a 5.65 MB model size, 2.85 M network parameters, and 5.6 G FLOPs. Compared to other detection algorithms such as YOLOv5, SSD, and Faster R-CNN, this method has a smaller model size, smaller network parameters, lower memory usage, lower computation usage, and faster detection speed while maintaining high detection performance.

Future research will explore more efficient and lightweight feature extraction modules for deep convolutional neural networks. This will enable the network model to better extract the intricate and variable characteristics of blueberry fruits.

Author Contributions: Conceptualization, F.X., H.W., Y.X. and Z.S.; data curation, F.X., H.W., Y.X. and Z.S.; formal analysis, F.X., H.W., Y.X. and Z.S.; funding acquisition, H.W.; investigation, F.X., H.W., Y.X. and Z.S.; methodology, F.X., H.W., Y.X. and Z.S.; project administration, F.X. and H.W.; resources, F.X., H.W., Y.X. and Z.S.; software, F.X. and Y.X.; supervision, H.W.; validation, Y.X. and Z.S.; visualization, F.X., Y.X. and Z.S.; writing—original draft, F.X; writing—review and editing, F.X. and H.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of Heilongjiang Province of China (LH2020C047) and the China Postdoctoral Science Foundation (2019T120248).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data presented in this study can be requested from the corresponding author. The data is not currently available for public access because it is part of an ongoing research project.

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

References

1. Krishna, P.; Pandey, G.; Thomas, R.; Parks, S. Improving Blueberry Fruit Nutritional Quality through Physiological and Genetic Interventions: A Review of Current Research and Future Directions. *Antioxidants* **2023**, *12*, 810. [CrossRef] [PubMed]
2. Xiao, F.; Wang, H.; Li, Y.; Cao, Y.; Lv, X.; Xu, G. Object Detection and Recognition Techniques Based on Digital Image Processing and Traditional Machine Learning for Fruit and Vegetable Harvesting Robots: An Overview and Review. *Agronomy* **2023**, *13*, 639. [CrossRef]
3. Wang, H.; Lv, X.; Xiao, F.; Sun, L. Analysis and Testing of Rigid–Flexible Coupling Collision Harvesting Processes in Blueberry Plants. *Agriculture* **2022**, *12*, 1900. [CrossRef]
4. Obsie, E.Y.; Qu, H.; Zhang, Y.J.; Annis, S.; Drummond, F. Yolov5s-CA: An Improved Yolov5 Based on the Attention Mechanism for Mummy Berry Disease Detection. *Agriculture* **2023**, *13*, 78. [CrossRef]
5. Yang, W.; Ma, X.; Hu, W.; Tang, P. Lightweight Blueberry Fruit Recognition Based on Multi-Scale and Attention Fusion NCBAM. *Agronomy* **2022**, *12*, 2354. [CrossRef]
6. Yang, W.; Ma, X.; An, H. Blueberry Ripeness Detection Model Based on Enhanced Detail Feature and Content-Aware Reassembly. *Agronomy* **2023**, *13*, 1613. [CrossRef]
7. Wang, H.; Feng, J.; Yin, H. Improved Method for Apple Fruit Target Detection Based on YOLOv5s. *Agriculture* **2023**, *13*, 2167. [CrossRef]
8. Gu, B.; Wen, C.; Liu, X.; Hou, Y.; Hu, Y.; Su, H. Improved YOLOv7-Tiny Complex Environment Citrus Detection Based on Lightweighting. *Agronomy* **2023**, *13*, 2667. [CrossRef]
9. Ren, R.; Sun, H.; Zhang, S.; Wang, N.; Lu, X.; Jing, J.; Xin, M.; Cui, T. Intelligent Detection of Lightweight "Yuluxiang" Pear in Non-Structural Environment Based on YOLO-GEW. *Agronomy* **2023**, *13*, 2418. [CrossRef]
10. Xiong, J.; Liu, Z.; Chen, S.; Liu, B.; Zheng, Z.; Zhong, Z.; Yang, Z.; Peng, H. Visual Detection of Green Mangoes by an Unmanned Aerial Vehicle in Orchards Based on a Deep Learning Method. *Biosyst. Eng.* **2020**, *194*, 261–272. [CrossRef]
11. Zhang, W.; Wang, J.; Liu, Y.; Chen, K.; Li, H.; Duan, Y.; Wu, W.; Shi, Y.; Guo, W. Deep-Learning-Based in-Field Citrus Fruit Detection and Tracking. *Hortic. Res.* **2022**, *9*, uhac003. [CrossRef] [PubMed]
12. Gao, F.; Fang, W.; Sun, X.; Wu, Z.; Zhao, G.; Li, G.; Li, R.; Fu, L.; Zhang, Q. A Novel Apple Fruit Detection and Counting Methodology Based on Deep Learning and Trunk Tracking in Modern Orchard. *Comput. Electron. Agric.* **2022**, *197*, 107000. [CrossRef]
13. Miao, Z.; Yu, X.; Li, N.; Zhang, Z.; He, C.; Li, Z.; Deng, C.; Sun, T. Efficient Tomato Harvesting Robot Based on Image Processing and Deep Learning. *Precis. Agric.* **2023**, *24*, 254–287. [CrossRef]
14. Yu, Y.; Zhang, K.; Yang, L.; Zhang, D. Fruit Detection for Strawberry Harvesting Robot in Non-Structural Environment Based on Mask-RCNN. *Comput. Electron. Agric.* **2019**, *163*, 104846. [CrossRef]
15. Jia, W.; Tian, Y.; Luo, R.; Zhang, Z.; Lian, J.; Zheng, Y. Detection and Segmentation of Overlapped Fruits Based on Optimized Mask R-CNN Application in Apple Harvesting Robot. *Comput. Electron. Agric.* **2020**, *172*, 105380. [CrossRef]

16. Li, J.; Zhou, H.; Jayas, D.S.; Jia, Q. Construction of a Dataset of Stored-Grain Insects Images for Intelligent Monitoring. *Appl. Eng. Agric.* **2019**, *35*, 647–655. [CrossRef]
17. Xiong, Z.; Wang, L.; Zhao, Y.; Lan, Y. Precision Detection of Dense Litchi Fruit in UAV Images Based on Improved YOLOv5 Model. *Remote Sens.* **2023**, *15*, 4017. [CrossRef]
18. Cai, D.; Lu, Z.; Fan, X.; Ding, W.; Li, B. Improved YOLOv4-Tiny Target Detection Method Based on Adaptive Self-Order Piecewise Enhancement and Multiscale Feature Optimization. *Appl. Sci.* **2023**, *13*, 8177. [CrossRef]
19. Bie, M.; Liu, Y.; Li, G.; Hong, J.; Li, J. Real-Time Vehicle Detection Algorithm Based on a Lightweight You-Only-Look-Once (YOLOv5n-L) Approach. *Expert Syst. Appl.* **2023**, *213*, 119108. [CrossRef]
20. Zhou, Z.; Fang, Z.; Wang, J.; Chen, J.; Li, H.; Han, L.; Zhang, Z. Driver Vigilance Detection Based on Deep Learning with Fused Thermal Image Information for Public Transportation. *Eng. Appl. Artif. Intell.* **2023**, *124*, 106604. [CrossRef]
21. Li, Y.; Xue, J.; Zhang, M.; Yin, J.; Liu, Y.; Qiao, X.; Zheng, D.; Li, Z. YOLOv5-ASFF: A Multistage Strawberry Detection Algorithm Based on Improved YOLOv5. *Agronomy* **2023**, *13*, 1901. [CrossRef]
22. Yu, G.; Zhou, X. An Improved YOLOv5 Crack Detection Method Combined with a Bottleneck Transformer. *Mathematics* **2023**, *11*, 2377. [CrossRef]
23. Yang, W.; Liu, T.; Jiang, P.; Qi, A.; Deng, L.; Liu, Z.; He, Y. A Forest Wildlife Detection Algorithm Based on Improved YOLOv5s. *Animals* **2023**, *13*, 3134. [CrossRef] [PubMed]
24. Niu, S.; Zhou, X.; Zhou, D.; Yang, Z.; Liang, H.; Su, H. Fault Detection in Power Distribution Networks Based on Comprehensive-YOLOv5. *Sensors* **2023**, *23*, 6410. [CrossRef] [PubMed]
25. 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 (CVPR 2018), Salt Lake City, UT, USA, 18–23 June 2018. [CrossRef]
26. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In Proceedings of the 15th European Conference on Computer Vision (ECCV 2018), Munich, Germany, 8–14 September 2018. [CrossRef]
27. Zhang, T.; Sui, Y.; Wu, S.; Shao, F.; Sun, R. Table Structure Recognition Method Based on Lightweight Network and Channel Attention. *Electronics* **2023**, *12*, 673. [CrossRef]
28. Wei, B.; Chen, H.; Ding, Q.; Luo, H. SiamAGN: Siamese Attention-Guided Network for Visual Tracking. *Neurocomputing* **2022**, *512*, 69–82. [CrossRef]
29. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the 15th European Conference on Computer Vision (ECCV 2018), Munich, Germany, 8–14 September 2018. [CrossRef]
30. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020), Seattle, WA, USA, 13–19 June 2020. [CrossRef]
31. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef]
32. Lu, A.; Ma, L.; Cui, H.; Liu, J.; Ma, Q. Instance Segmentation of Lotus Pods and Stalks in Unstructured Planting Environment Based on Improved YOLOv5. *Agriculture* **2023**, *13*, 1568. [CrossRef]