



A lightweight method for apple-on-tree detection based on improved YOLOv5

Mei Li^{1,2} · Jiachuang Zhang^{1,2} · Hubin Liu^{1,2} · Yuhui Yuan^{1,2} · Junhui Li^{1,2} · Longlian Zhao^{1,2}

Received: 19 February 2024 / Revised: 6 May 2024 / Accepted: 1 June 2024 / Published online: 14 June 2024
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2024

Abstract

After apple fruit maturation, the optimal harvest period is short, and the picking robot is expected to improve harvesting efficiency. While it is common for apples to be overlapped and occluded by branches and leaves, which pose challenges to the robot's apple harvesting. Therefore, precise and swift identification and localization of the target fruit is crucial. To this end, this paper proposes a lightweight apple detection method, YOLOv5s-ShuffleNetV2-DWconv-Add, or “YOLOv5s-SDA” for short. The red and green apple datasets in natural environment were collected by a mobile phone, which were divided into four categories: red and green apples that can be directly grasped and cannot be directly grasped, in order to avoid damage to the robotic arm. Different deep learning object detection models were compared, with the YOLOv5s algorithm providing superior recognition performance. To improve harvest efficiency and portability of hardware devices, modifications are made to the YOLOv5s algorithm, replacing the Focus, C3, and Conv structures within the backbone with 3×3 Conv structures and ShuffleNetV2, removing SPP and C3 structures; substituting the C3 in the Neck portion with DWConv modules; and replacing two Concat layers in the PANet structure with smaller computational Add layers. Results demonstrate that the model achieved a mAP of 94.6% on the test set, doubled the detection speed, and compressed the model weight to 11.8% of its original value, while maintaining model accuracy. This new method exhibits promising performance in fruit target recognition in natural scenes, providing an effective means of visual acquisition for fruit picking robots.

Keywords Apple detection · YOLOv5 · Lightweight · Deep learning · Complex environment

1 Introduction

Apples are widely popular fruits with a broad market for cultivation and consumption. The optimal picking period for ripe fruits is short, and manual harvesting is inefficient. Consequently, there is an urgent need to introduce intelligent apple-picking robots that can revolutionize the harvesting process and enhance efficiency.

In recent years, the rapid advancements in robotics technology [1], machine vision techniques [2], and artificial

intelligence [3–5] have provided opportunities for the development of intelligent agricultural machinery and smart orchard systems. Consequently, there has been a growing body of research focused on the development of apple-picking robots suitable for natural environments. Orchards present complex settings, characterized by foliage occlusion and overlapping. These intricate backgrounds can interfere with the robot's ability to accurately grasp the fruits. The emergence of deep learning-based object detection techniques, which leverage large-scale datasets for modeling, offers the potential for precise and efficient fruit detection in complex backgrounds. These techniques demonstrate robust data representation and feature extraction capabilities, enabling them to tackle object recognition challenges in complex scenarios while meeting requirements for both accuracy and real-time performance. As a result, they have been widely applied in the agricultural domain [6] including fruit detection [7, 8] crop disease diagnosis [9] and other applications.

✉ Longlian Zhao
zhaolonglian@aliyun.com

¹ College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, China

² Key Laboratory of Modern Precision Agriculture System Integration Research, Ministry of Education, Beijing 100083, China

Deep learning object detection algorithms primarily consist of two types. The first type is the two-stage object detection technique represented by Faster R-CNN [10], which involves region proposal generation followed by classification and regression. The second type is the one-stage object detection technique that directly predicts regression results without generating region proposals. It includes the YOLO series and SSD [11], which have made significant progress in fruit detection in recent years. Yu et al. [12] employed a mask region convolutional neural network (Mask R-CNN) with ResNet50 as the backbone network for feature extraction. This approach achieved accurate strawberry recognition in different lighting conditions, with adhesion, overlap, occlusion, and other complex growth states. The average detection rate on a test set of 100 images reached 97.78%, demonstrating excellent generality and robustness. Wan et al. [13] improved the Faster R-CNN model to detect multiple types of fruits, such as apples, mangoes, and oranges, achieving higher detection accuracy and faster processing speed.

The YOLO model [14] emerged in 2016 and quickly gained widespread application in real-time object detection due to its advantage of high speed. Given the requirement for real-time performance in fruit harvesting, single-stage object detection algorithms have gradually been applied in fruit detection in recent years. Zhao et al. [15] proposed an apple localization method based on the YOLOv3 deep convolutional neural network. This approach enables fruit recognition and localization under different lighting conditions, accounting for occlusion, adhesion, and bagging. It enhances the efficiency and adaptability of apple-picking robots in various scenarios. Xuan et al. [16] proposed an enhanced YOLOv3 model, which was compared with the Faster RCNN model based on AlexNet, the Faster RCNN model based on ResNet101, and the YOLOv3 model based on DarkNet53. Experimental results demonstrated that the improved model achieved higher recognition accuracy. Peng et al. [17] introduced an improved SSD model, capable of recognizing multiple types of fruits. Chen et al. [18] enhanced the YOLOv4 model by incorporating the DenseNet module, improving the feature extraction network. As a result, the modified YOLOv4 model exhibited superior detection performance, with an average recognition accuracy of 97.13%. Lv et al. [19] made improvements to the YOLOv5 model by employing the ACON-C activation function, achieving a mAP of 98.4%. Consequently, it could detect apples with different growth forms.

Fruits can grow in various poses in natural environments, making them vulnerable to lighting interference. To improve detection accuracy, deep learning object detection network models have continuously evolved, resulting in increasing model parameter and computational complexity,

posing challenges to model deployment and portability [20, 21]. Therefore, research on model lightweighting through reducing model parameters and computational complexity is highly important while achieving high accuracy. Wang et al. [22] pruned the YOLOv5s detection model using a channel pruning method and fine-tuning techniques. The compressed model was reduced to 10% of its original size, while maintaining detection accuracy for small apples and improving detection speed. Ji et al. [23] replaced the backbone network of YOLOX with the lightweight ShuffleNetv2 network, added a Convolutional Block Attention Module (CBAM), and introduced an Adaptive Spatial Feature Fusion (ASFF) module. The modified model achieved an average detection accuracy of 96.67% and FPS of 65, with excellent detection speed and accuracy. Xu et al. [24] utilized the lightweight GhostNet as the backbone network and introduced the Mish activation function in the Neck network of YOLOv4 to reduce model parameters. Additionally, a channel attention mechanism (ECA) was adopted to enhance the multi-scale features of the backbone network. As a result, the accuracy improved by 2.26%, and the model size reduced from 250.7 MB to 43.5 MB.

In natural orchard environments, achieving high accuracy and efficiency in fruit recognition is made possible by deep learning object detection algorithms. These algorithms adaptively learn fruit features in different environmental conditions, enabling the identification of target fruits in complex scenarios. Compared to traditional learning algorithms, they exhibit significantly higher recognition efficiency and accuracy, while demonstrating strong model robustness and stability. However, the pursuit of high-precision fruit detection has led to the deepening of network structures, thereby compromising real-time performance. Considering the need for model portability on hardware devices, it is crucial to focus on model lightweighting, which involves maintaining accuracy while improving recognition speed and reducing parameters and computational complexity [25, 26]. This research primarily explores the application of deep learning object detection algorithms in apple recognition and detection under complex scenarios. By comparing different deep learning object detection models, the optimal fruit detection model is identified. Furthermore, lightweight improvements are made to the YOLOv5s model, resulting in enhanced detection speed while preserving accuracy.

2 Materials and methods

2.1 Data collection and preprocessing

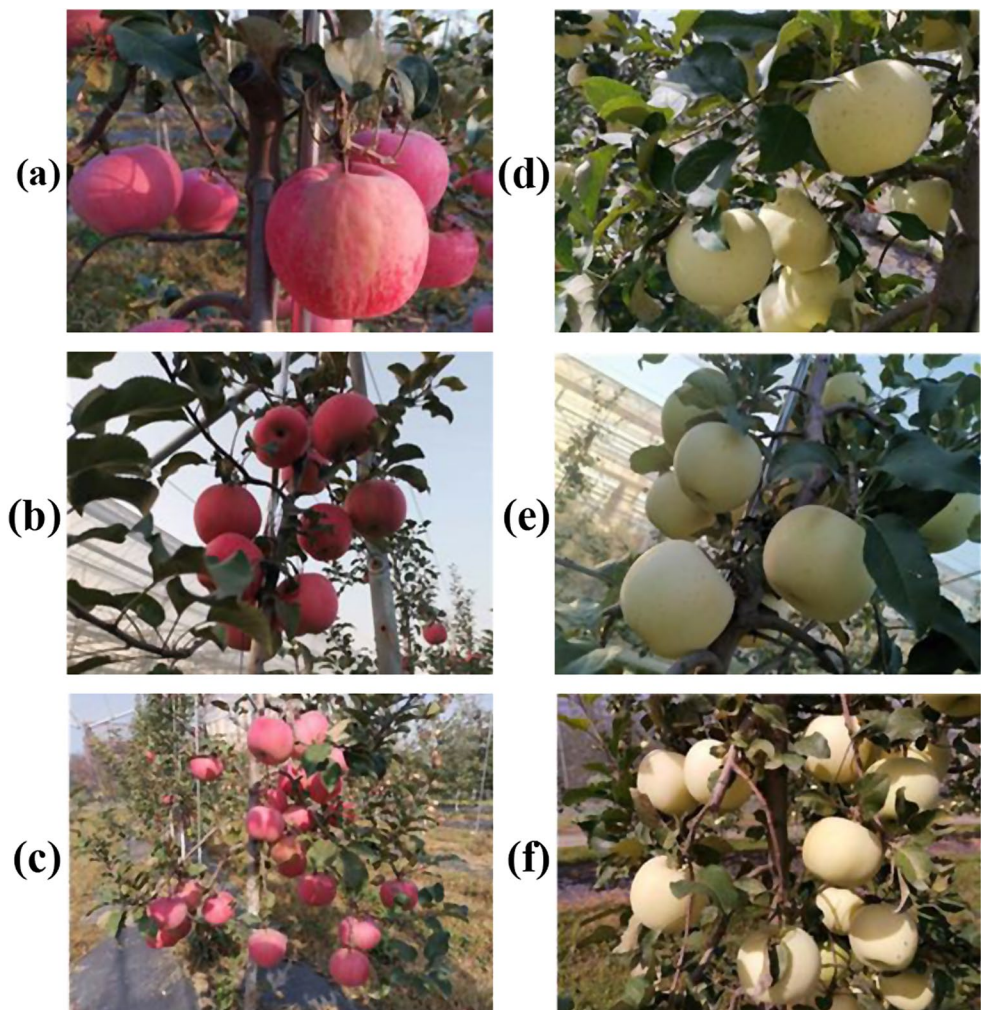
2.1.1 Data acquisition

The images in the dataset were captured in October 2021. Different images of green apples and red apples were collected at the Beijing Academy of Forestry and Pomology, Chinese Academy of Agricultural Sciences. Image collection took place on sunny mornings and afternoons. The Honor V20 smartphone was employed as the image capture device, with a resolution of 4000×3000 pixels, saving images in JPG format. A total of 4613 images were captured from various angles, including front-lit fruits, backlit fruits, fruits partially occluded by leaves, and fruit overlapping, representing natural apple fruit images in their environmental context. Some sample images from the dataset are illustrated in Fig. 1.

2.1.2 The dataset annotations and dataset partitioning

Due to the diverse morphology of apples, significant occlusions and overlapping can pose challenges during harvesting, potentially resulting in damage to mechanical hands. Existing studies tend to categorize fruit recognition into a single type, without specific consideration for occlusions caused by leaves, branches, or overlapping fruits. This oversight can easily lead to potential harm to the mechanical hand during fruit retrieval and hinder the planning of optimal fruit grasping paths [27–29]. Based on on-site observations of orchard picking environments and the picking status of robotic hands, apples were categorized into two main groups: directly grabbable and non-grabbable. To further distinguish between red and green apples, the apples were divided into four categories: directly grabbable red apples, directly grabbable green apples, non-grabbable red apples, and non-grabbable green apples. The dataset classification rules are as follows: fruits that are not occluded or only occluded by leaves are classified as directly grabbable, while fruits occluded by other fruits, branches, wires, steel

Fig. 1 The images of the dataset. (a) Red apples toward the light. (b) Backlit red apples. (c) Shadowed red apples. (d) Green apples toward the light. (e) Backlit green apples. (f) Shadowed green apples



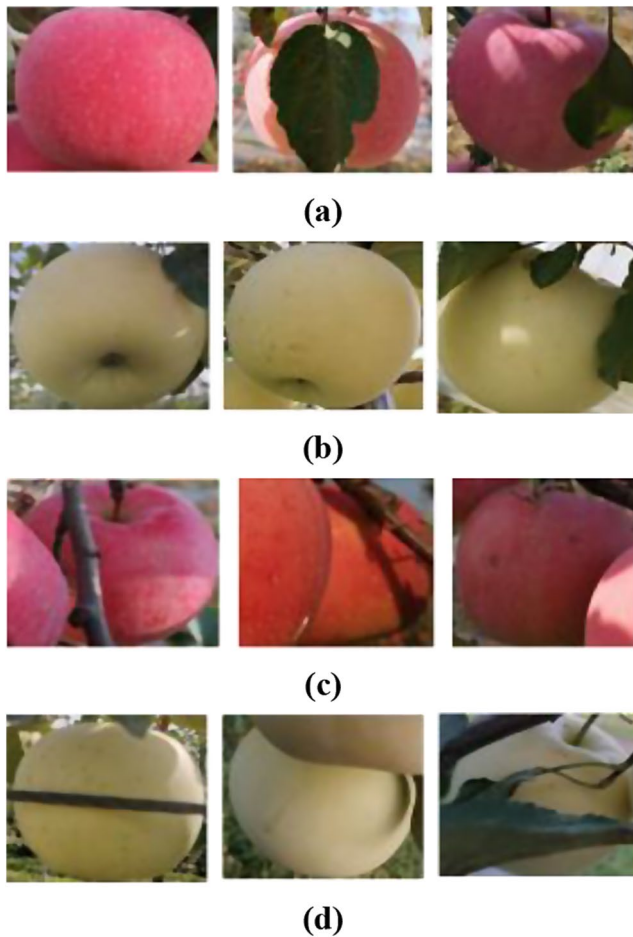


Fig. 2 Dataset categorization. (a) Red apples that can be grabbed directly. (b) Green apples that can be grabbed directly. (c) Red apples that cannot be grabbed directly. (d) Green apples that cannot be grabbed directly

Table 1 Dataset splitting

Training set /images	Validation set /images	Testing set /images
2767	923	923

pipes, or any other obstacles that may damage the robotic hand are classified as non-grabbable. This includes situations where the edges of the fruit, as well as one or more parts (top, bottom, left, right), are occluded. The classification is shown in Fig. 2.

All the apple images in the dataset were annotated using the Labellmg software, resulting in four categories:

“rappleyes” for directly graspable red apples, “roccludedno” for non-directly graspable red apples, “gappleyes” for directly graspable green apples, and “goccludedno” for non-directly graspable green apples. The class names were added to the top right corner of each rectangular bounding box, and the annotation files were saved in the XML format. The dataset was divided into training, validation, and testing sets in a ratio of 6:2:2, as shown in Table 1.

2.2 The principles of deep learning object detection methods

2.2.1 YOLOv5s network architecture

Considering the high real-time and accuracy requirements for apple harvesting, various deep learning object detection algorithms are compared, and the single-stage object detection algorithm YOLOv5, which has good real-time performance, is selected. Moreover, several lightweight improvement methods are attempted to establish the detection model for rapid and accurate apple detection. The YOLOv5 model is mainly divided into four types: YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, with increasing model sizes and model parameters as shown in Table 2.

From Table 2, it is evident that YOLOv5s has the smallest parameter size compared to the other three models, and also the fastest detection time. However, in terms of model prediction accuracy, these four models exhibit high similarity, all exceeding 95%. Considering the real-time requirement of apple harvesting and the need for lightweight embedding in hardware devices, this study selects the YOLOv5s model with fewer layers and parameters for apple object recognition and detection.

The structure of the YOLOv5s model is illustrated in Fig. 3, which mainly consists of the input, backbone network, neck, and prediction output layers [30]. Its working principle involves inputting an image and utilizing the Mosaic method for random expansion to achieve data augmentation. The model employs adaptive anchor box computation and adaptive image scaling techniques to mitigate distortions caused by varying image sizes. The backbone portion of the YOLOv5s model is primarily responsible for feature extraction and includes modules such as Focus, CBS, C3, and Spatial Pyramid Pooling (SPP). The Focus

Table 2 Indicator parameters of the four models of YOLOv5

Model	Depth	Width	Number of layers	Parameters	mAP @0.5	Weight files/MB	GFLOPs	T/ms
YOLOv5s	0.33	0.50	191	7.3 M	95.5%	14.4	16.4	20.7
YOLOv5m	0.67	0.75	263	21.4 M	95.3%	42.5	50.3	38.0
YOLOv5l	1.00	1.00	335	47.0 M	95.0%	93.8	114.0	72.0
YOLOv5x	1.33	1.25	407	87.7 M	95.5%	175.1	216.9	138.0

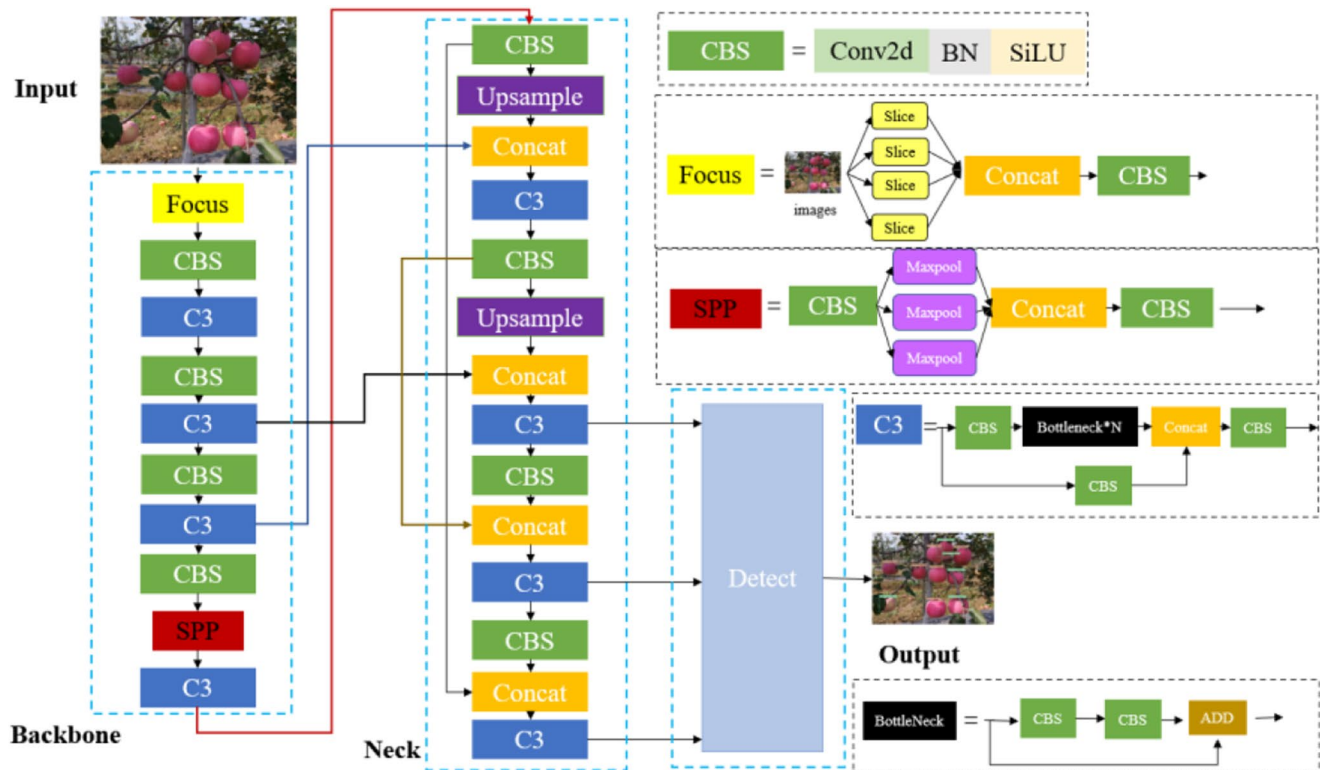


Fig. 3 The network architecture diagram of YOLOv5s

module is utilized to process image slicing, dividing the input image size of $640 \times 640 \times 3$ into a reduced feature map of size $320 \times 320 \times 12$. The module then employs Concatenation to combine the sliced feature maps. The goal of using this slicing method is to decrease network parameters and ensure that down-sampling information is not lost. The CBS module consists of convolution, Batch Normalization (BN), and SiLU activation function. The C3 module includes three standard CBS modules and several Bottleneck modules. It mainly comprises two branches: the first branch undergoes multiple Bottleneck stacking and one CBS module, while the second branch goes through another CBS module. Finally, the results of both branches are concatenated and processed by a CBS module. The SPP structure refers to applying three different convolutional kernels for maximum pooling (Maxpool) operations on the feature map. The output results are then concatenated to better fuse features and increase receptive fields. The Neck network of YOLOv5s adopts a Feature Pyramid Network (FPN) and a bottom-up Path Aggregation Network (PAN) structure. This helps in integrating features from different scales. Ultimately, the feature map is used for prediction, generating bounding boxes and predicted categories. To achieve optimal prediction results, non-maximum suppression (NMS) is used to remove redundant boxes. Finally, the predicted boxes,

confidence scores, and predicted categories are displayed in the image, completing the task of object recognition.

2.2.2 The lightweight improvements of YOLOv5s

Given the consideration that the algorithm will be transplanted onto embedded devices, the computational complexity, weight file size, and detection speed of the model are critical indicators. Hence, the following section focuses on the lightweight improvements of YOLOv5s. Common lightweight network models such as MobileNetV3 [31], GhostConv [32], and ShuffleNetV1-2 [33, 34] have been explored extensively. Experimental results reveal that, for the dataset employed in this study, the ShuffleNetV2 network model exhibits superior detection performance. This section will delve into further discussion and research based on the ShuffleNetV2 network model.

The backbone of the YOLOv5s model comprises the Focus, C3, and CBS modules. In this study, the Focus structure is replaced by the CBRM structure to mitigate the loss of sampling information and computational overhead. The CBRM structure is composed of Convolution, Batch Normalization, and Rectified Linear Unit (ReLU) activation function, eliminating the Slice operation in Focus and accelerating the training speed. Furthermore, the original C3 and CBS modules are substituted with the ShuffleNetV2

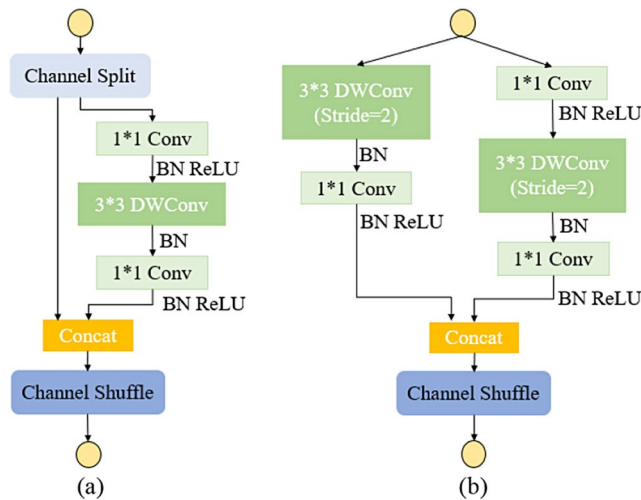


Fig. 4 The network architecture diagram of ShuffleNetV2. (a) The basic unit structure. (b) The spatial down sampling structure

network model. The excessive consumption of cache space and subsequent decrease in runtime speed are associated with the frequent use of the C3 module’s depth-wise separable convolution. To strike a balance between computational precision and speed, the lightweight ShuffleNetV2 network model is adopted, composed of alternating patterns of Fig. 4(a) and (b).

From Fig. 4(a), it can be observed that after the Channel Split operation, the network branch is divided into two branches. One branch undergoes two convolutional layers and a Depthwise Convolution (DWConv) layer before being concatenated using Concat. This ensures an output with the same channel width, reducing memory access space and improving computational speed. On the other hand, Fig. 4(b) removes the Channel Split operation, resulting in

an output channel twice the size of the input channel. The application of the ShuffleNetV2 network model effectively reduces model parameters and accelerates computational speed.

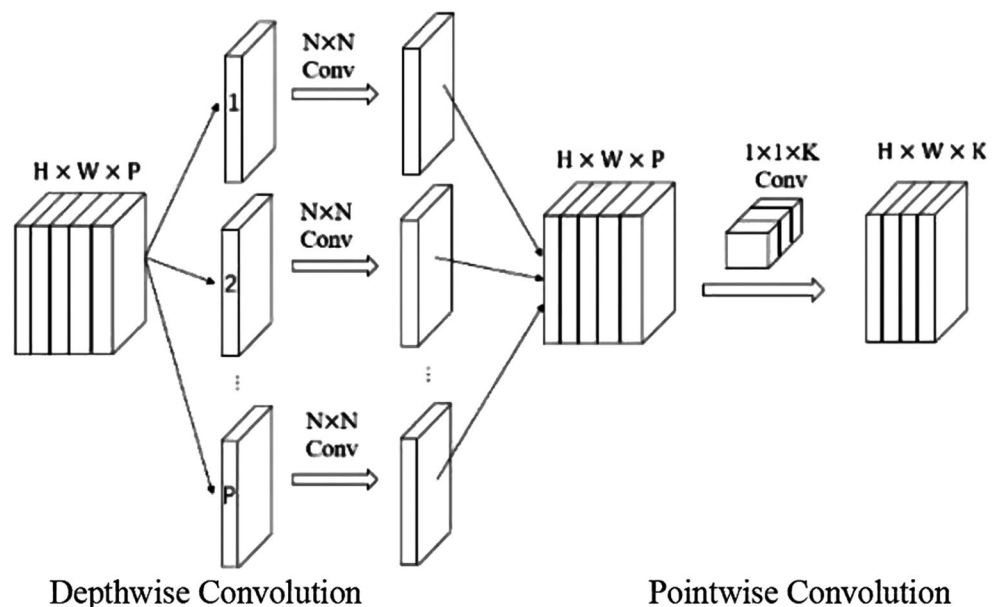
DWConv [35] is a lightweight convolution proposed in 2017, which differs from conventional convolutions as it involves two steps: depthwise convolution and pointwise convolution. Traditional convolutions operate by convolving each channel of an input image of size $H \times W \times P$ with $K \times N \times N$ convolution kernels when the stride is 1. The computational complexity is $H \times W \times P \times K \times N \times N$. In contrast, Depthwise Separable Convolution divides the process into two steps, as illustrated in Fig. 5.

The process first involves a depthwise convolution on the input image of size $H \times W \times P$ using $P \times N \times N$ convolution kernels, followed by the use of a $1 \times 1 \times K$ convolution kernel. The total computational complexity is $H \times W \times P \times N \times N + H \times W \times P \times K$. By comparing the computational complexity of depthwise and traditional convolutions, the ratio between the two, denoted as S , is obtained and can be expressed as follows.

$$S = \frac{H \times W \times P \times N \times N + H \times W \times P \times K}{H \times W \times P \times K \times N \times N} = \frac{1}{K} + \frac{1}{N^2} \tag{1}$$

From the equation, it can be observed that the computational complexity of DWConv is compressed to $1/K + 1/N^2$ times compared to the original convolution. It is evident that DWConv can effectively reduce the computational burden of model parameters.

Fig. 5 Separable depthwise convolution diagram



In order to further reduce memory usage and accelerate computation speed, improvements are made to the Neck section of YOLOv5s. This involves adjusting the input and output of all layers to have the same number of channels and replacing all C3 layers in the Neck with DWConv modules that have lower computational complexity. This effectively reduces parameter computations. To further reduce computational burden, the original Concat connections in the PAN feature fusion section are replaced with Add connections. Both Concat and Add are methods of feature fusion, where Concat merges information from output layers, increasing the number of channels without altering the information content. On the other hand, Add keeps the number of channels unchanged while increasing the amount of information. By replacing Concat with Add, computational complexity is reduced. Based on these improvements, the modified lightweight model YOLOv5s-ShuffleNetV2-DWconv-Add (YOLOv5s-SDA) is constructed, as depicted in Fig. 6.

2.3 Experimental setup and evaluation indicators

2.3.1 Experimental equipment and parameter settings

All training and testing experiments are conducted on the same server with a computer configuration of 43G Intel(R) Xeon(R) Platinum 8255C and NVIDIA GeForce RTX 2080Ti graphics card, running Ubuntu18.04LTS as

its operating system. Model training is carried out using the PyTorch framework. All network models are trained based on pretrained models. After training, the best model is selected as the prediction model for testing on the test set, with all predictions performed on a desktop NVIDIA GeForce RTX 1050Ti.

2.3.2 Evaluation indicators

To comprehensively evaluate the performance of the model, precision (P), recall (R), average precision (AP), mean average precision (mAP), and F1 score are employed as evaluation indicators for model accuracy. Additionally, model parameters, size of weight file, floating-point operations per second (GFLOPs), and detection time per image are considered as evaluation indicators for model speed and resource consumption. The formulas are as follows:

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$AP = \int_0^1 P(R)dR \tag{4}$$

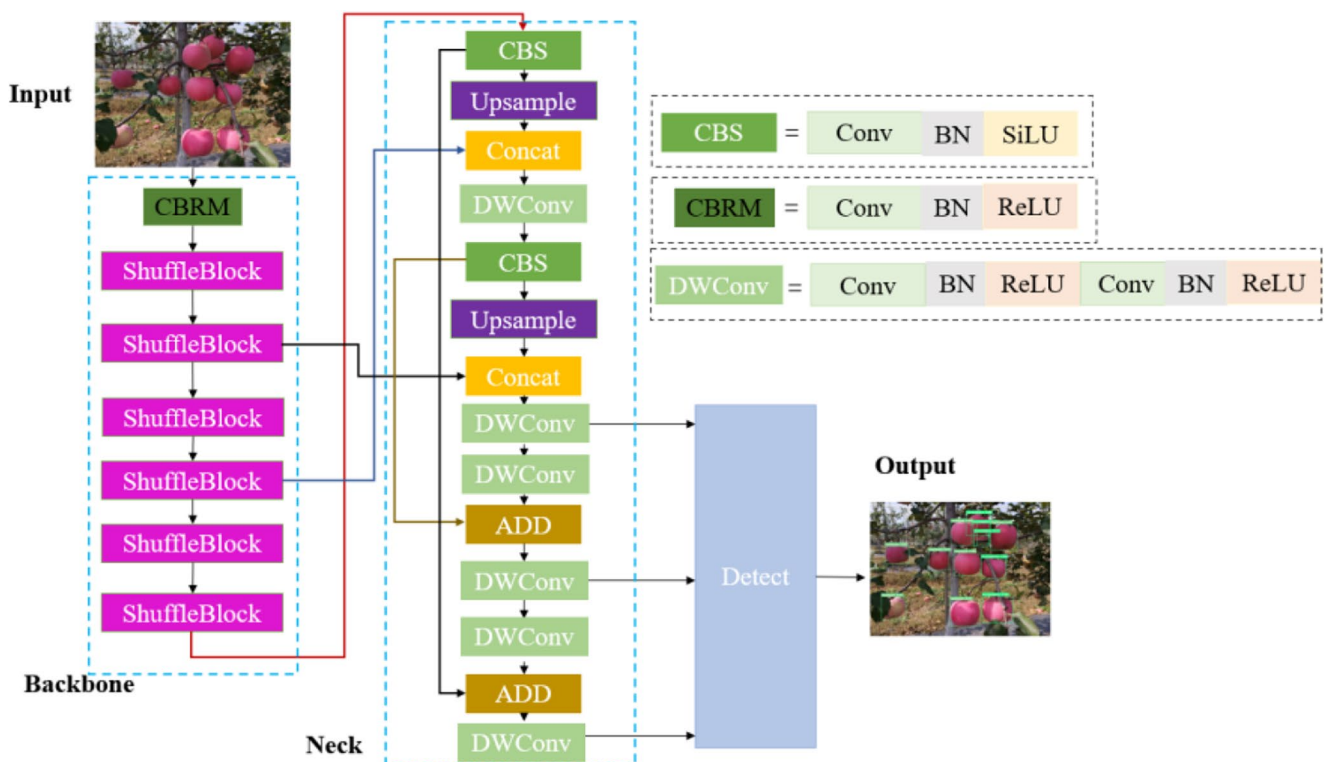
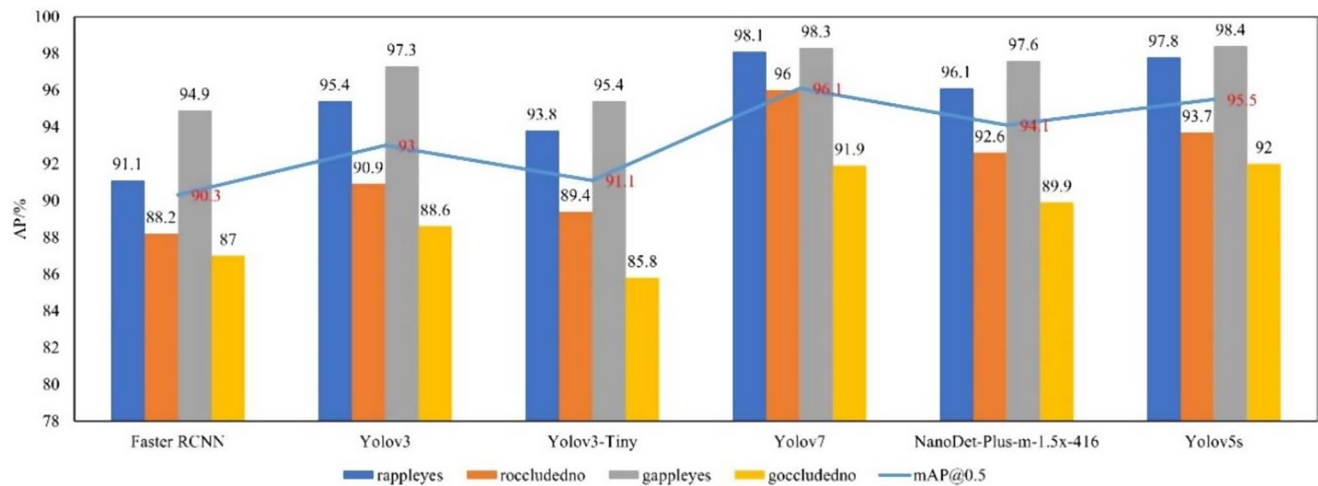


Fig. 6 The improved lightweight network of YOLOv5s-SDA

Table 3 Comparison of different deep learning detection algorithms

Model	mAP@0.5	F1	Parameters	Weight files/ MB	Single image detection time/ms
Faster RCNN	90.3%	85.2%	--	1126.4	164.2
YOLOv3	93.0%	86.5%	61.50 M	469.0	110.5
YOLOv3-Tiny	91.1%	84.9%	8.67 M	66.2	15.4
YOLOv7	96.1%	91.6%	70.80 M	135.0	167.7
NanoDet-Plus-m-1.5x-416	94.1%	--	2.44 M	16.4	20.5
YOLOv5s	95.5%	91.0%	7.10M	14.4	20.7

**Fig. 7** The recognition accuracy for four different fruit categories using different detection algorithms

$$mAP = \frac{\sum_{i=1}^n AP_i}{n} \quad (5)$$

$$F1 = \frac{2 \times P \times R}{P + R} \quad (6)$$

$$M = 0.1 \times mAP@0.5 + 0.9 \times mAP@0.5 : 0.95 \quad (7)$$

In Eqs. (2) and (3), TP represents the number of positive samples predicted as positive, FN denotes the number of positive samples predicted as negative, FP represents the number of negative samples predicted as positive. Additionally, mAP in Eq. (5) is calculated as the mean of the AP values across all categories.

3 Results and discussion

3.1 Comparison of different detection algorithms

To find the optimal apple recognition algorithm, the apple dataset was employed with various object detection algorithms, namely Faster RCNN, YOLOv3, YOLOv3-Tiny, YOLOv5s, NanoDet-Plus-m-1.5x-416 and YOLOv7, to

conduct comparisons. Faster RCNN underwent 35,200 iterations, while the other algorithms underwent 300 iterations. All models were compared on the same dataset, and the best-performing weight file obtained during training was used for testing. Finally, different detection algorithms were applied to predict on the test set, and the prediction results are presented in Table 3.

To visually compare the detection performance of different deep learning object detection methods, Fig. 7 presents a comparison of recognition accuracy for four different categories of fruit, namely, “rappleyes” representing directly grappable red apples, “roccludedno” representing ungrappable red apples, “gappleyes” representing directly grappable green apples, and “goccludedno” representing ungrappable green apples. From Fig. 7, it is evident that YOLOv7 and YOLOv5s algorithms exhibit the best fruit recognition performance, with high recognition accuracy for all categories.

Considering the future application of the apple recognition algorithm in orchard picking robots for real-time fruit harvesting, factors such as recognition accuracy, model size, and detection speed are vital indicators that need to be comprehensively evaluated. From Table 3, it is evident that the Faster RCNN algorithm has the lowest mAP and larger weight files, indicating more space occupation during future transplantation. On the other hand, the YOLOv7 algorithm

exhibits the highest mAP and F1 score, yielding the best recognition results. The lightweight object detection algorithm NanoDet-Plus-m-1.5x-416 has a weight file size and detection time similar to YOLOv5s, but its accuracy appears to be inferior to YOLOv5s as shown in Fig. 7. However, the YOLOv7 network model is excessively large, with model parameters reaching 70.80 M, which is an order of magnitude higher compared to other algorithms. Additionally, the YOLOv7 algorithm has the longest single-image detection time, resulting in slower recognition speed. Taking these factors into account, it is not the optimal fruit recognition algorithm. In comparison, the YOLOv5s algorithm demonstrates slightly lower average recognition accuracy and decreased F1 score compared to YOLOv7. However, it outperforms Faster RCNN, YOLOv3, YOLOv3-Tiny and NanoDet-Plus-m-1.5x-416 algorithms. Furthermore, the YOLOv5s algorithm exhibits relatively faster single-image detection time, with a recognition speed eight times faster than YOLOv7. This showcases favorable performance. Overall, among these algorithms, the YOLOv5s algorithm displays the best performance and detection effectiveness. Hence, the YOLOv5s model is selected as the recognition algorithm for further research.

3.2 Comparative analysis of different lightweight backbone network models

In order to investigate a more lightweight algorithm with smaller model weight files, fewer parameters, and faster detection speed, this section aims to compare the prediction performance of different lightweight models based on YOLOv5s.

The YOLOv5s architecture is modified by replacing its original backbone networks with more lightweight network structures, namely MobileNetV3, GhostConv, and ShuffleNetV2, resulting in YOLOv5s-MobileNetV3 (abbreviated as YOLOv5s-Mv3) and YOLOv5s-ShuffleNetV2 (abbreviated as YOLOv5s-Sv2). These modified models are then compared with the lightweight model YOLOv5s-ShuffleNetV2-DWconv-Add (abbreviated as YOLOv5s-SDA) as described in Sect. 2.2.2. During the training process, the

model automatically saved the model with the highest M value, which is then used as the optimal model for prediction (computed according to Eq. (7)). The generated weight files are utilized to predict the sample images in the test dataset, and the detection results of different lightweight backbone network models on the same test dataset are presented in Table 4.

Table 4 presents the recognition results of different lightweight network models, including mAP@0.5, AP for each class, F1 score, model parameters, weight file size, floating-point operation GFLOPs, and model single-image detection time. From Table 4, it can be observed that the utilization of lightweight network models with fewer parameters and shallower networks leads to a slight decrease in mAP and AP for each class. Specifically, when comparing YOLOv5s-MobileNetV3, YOLOv5s-GhostConv, and YOLOv5s-ShuffleNetV2, the mAP values decreased by 0.8%, 0.6%, and 0.9%, respectively, and the F1 scores decreased by 1.4%, 0.9%, and 1.5%, respectively. In terms of recognition accuracy, the YOLOv5s-GhostConv model exhibited the best precision. Moreover, for the evaluation of model speed, the model parameters, weight files, GFLOPs, and single-image detection times for the three models decreased. Specifically, the weight files of YOLOv5s-MobileNetV3, YOLOv5s-GhostConv, and YOLOv5s-ShuffleNetV2 were compressed to 69.4%, 54.5%, and 53.1%, respectively, the GFLOPs were reduced to 68.9%, 51.2%, and 48.8% of the original, and the single-image detection times were decreased to 95.2%, 88.4%, and 70.0% of the original, respectively. Considering the detection speed and portability, the YOLOv5s-ShuffleNetV2 model had better performance. Although the mAP of the YOLOv5s-GhostConv model was 0.3% higher than that of the YOLOv5s-ShuffleNetV2 model, its detection speed was slower by 3.8 ms. Therefore, the final decision was made to use ShuffleNetV2 as the backbone network and improve it based on the YOLOv5s-ShuffleNetV2 model.

The improved lightweight model, YOLOv5s-ShuffleNetV2-DWconv-Add (YOLOv5s-SDA), also demonstrated its detection results on the prediction set, as shown in Table 4. From the table, it can be observed that compared to the YOLOv5s-ShuffleNetV2 model, the YOLOv5s-SDA

Table 4 Comparative analysis of detection results using different lightweight backbone network algorithms

Network model	mAP@0.5/%	AP@0.5/%				F1/%	Parameters	Weight files /MB	GFLOPs	T /ms
		ry	rn	gy	gn					
YOLOv5s	95.5	97.8	93.7	98.4	92.0	91.0	7.10 M	14.4	16.4	20.7
YOLOv5s-Mv3	94.7	97.0	93.4	98.0	90.4	89.6	5.00 M	9.99	11.3	19.7
YOLOv5s-Gcv	94.9	97.1	93.0	98.3	91.1	90.1	3.80 M	7.85	8.4	18.3
YOLOv5s-Sv2	94.6	96.8	93.6	98.0	90.2	89.5	3.79 M	7.64	8.0	14.5
YOLOv5s-SDA	94.6	96.9	92.7	98.0	90.9	89.2	0.72 M	1.7	2.5	9.5

Note In the table, “ry” represents “rapplees,” indicating directly graspable red apples; “rn” represents “rocludedno,” indicating non-directly graspable red apples; “gy” represents “gapplees,” indicating directly graspable green apples; “gn” represents “goccludedno,” indicating non-directly graspable green apples; “T” represents the detection time for a single image

Table 5 Ablation experiments

Sv2	D	A	mAP@0.5%	AP@0.5%				F1/%	Parameters	Weight files /MB	GFLOPs	T /ms
				ry	m	gy	gn					
✓			95.5	97.8	93.7	98.4	92.0	91.0	7.10 M	14.4	16.4	20.7
			94.6	96.8	93.6	98.0	90.2	89.5	3.79 M	7.64	8.0	14.5
	✓		95.3	97.5	94.1	98.1	91.6	90.3	4.99 M	10.2	12.6	13.5
		✓	95.6	97.7	94.4	98.3	92.0	90.4	6.90 M	14.1	16.1	15.7
✓			94.5	97.2	92.8	97.8	90.2	88.8	0.74 M	1.7	2.6	14.4
		✓	94.8	97.4	93.2	98.3	90.4	90.5	4.83 M	9.9	12.4	14.5
✓		✓	94.3	96.9	92.0	97.8	90.5	88.8	0.98 M	2.2	3.4	17
✓		✓	94.6	96.9	92.7	98.0	90.9	89.2	0.72 M	1.7	2.5	9.5

Note In the table, “Sv2” represents “ShuffleNetV2,” “D” represents “DWConv,” “A” represents “ADD”

model maintained the same mAP, with the AP values for each category remaining relatively stable. However, there is a slight decrease of 0.3% in F1 score for the YOLOv5s-SDA model. Remarkably, the model exhibited improved speed, with model parameters reduced by an order of magnitude. Furthermore, the weight file size decreased to 22.3% of its original size, and the GFLOPs also saw a significant decrease. Notably, the detection time for a single image became 5ms faster than before, indicating that the model has become more lightweight.

To provide a more comprehensive display of the improved model, a comparison was made between the lightweight YOLOv5s-SDA model and the original YOLOv5s model. Despite a slight decrease in mAP from 95.5 to 94.6%, as well as a decline of 1.8% in F1 score, notable improvements were observed. The model parameters were reduced by an order of magnitude, with the weight file size shrinking from the original 14.4 MB to 1.7 MB, a reduction to 11.8%. Furthermore, the GFLOPs decreased from 16.4 to 2.5, representing only 15.2% of the original floating-point operations. The detection time for a single image also decreased from 20.7ms to 9.5ms, resulting in a detection speed that was 2.2 times faster than before. Considering the future deployment of the recognition algorithm on embedded devices, combined with a robotic arm to achieve real-time apple grasping, the proposed improvement method not only maintained a slightly decreased accuracy but also enhanced the detection speed. Additionally, it effectively compressed the model size, reduced the weight file size and model parameters, thereby occupying less memory and exhibiting desirable portability.

From Table 5, it can be observed that individually adding ShuffleNetV2, DWConv, and ADD modules did not reduce the order of magnitude of the model parameters. However, incorporating ShuffleNetV2 into other modules reduced the model parameters by an order of magnitude. This is because ShuffleNetV2 utilizes techniques such as depth-wise separable convolution and channel shuffle to reduce the number of parameters and computational costs, thereby decreasing network parameters without significantly affecting model accuracy. Combining models can reduce parameters while improving detection speed and decreasing model size. As shown in the table, simultaneously adding ShuffleNetV2 and DWConv to YOLOv5s reduced the model size to 11.8% of the original, with a detection time faster by 6.3ms; adding ShuffleNetV2 and ADD reduced the model size to 15.3% of the original, with a detection time faster by 3.7ms; adding DWConv and ADD reduced the model size to 68.8% of the original, with a detection time faster by 6.2ms, indicating that DWConv can improve model detection speed. Finally, combining the original YOLOv5s model with ShuffleNetV2, DWConv, and ADD, while maintaining accuracy, yielded

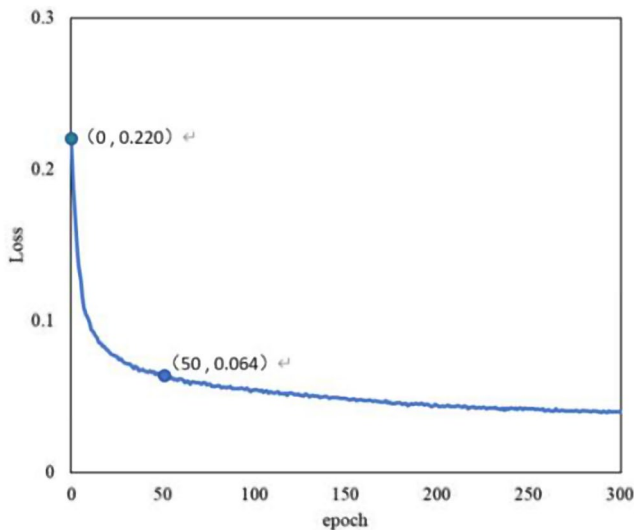


Fig. 8 Variation of the loss function during the training process

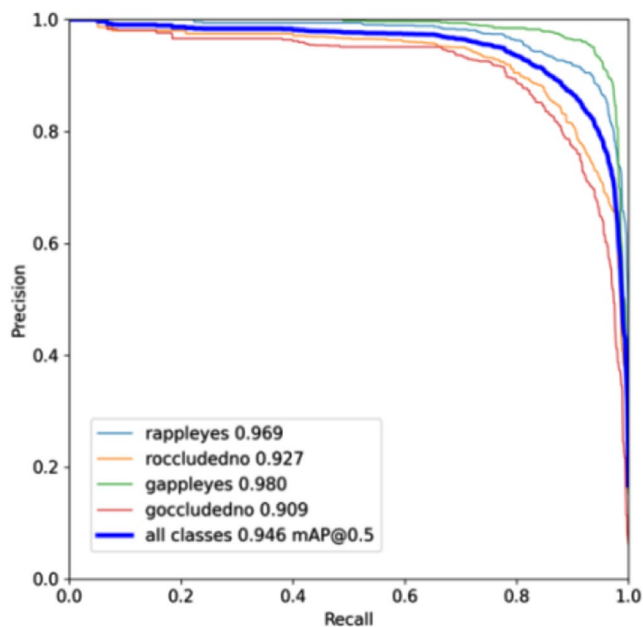


Fig. 9 YOLOv5s-SDA algorithm's P-R curve

better results in terms of model parameters, size, detection time, and other metrics compared to other combinations.

3.3 The experimental results under different natural conditions

To evaluate the training results and determine if the YOLOv5s-SDA model converged, Fig. 8 illustrates the loss function curve during the training process. The loss function rapidly decreased from 0.220 to 0.064 within the first 50 epochs, and gradually stabilized around 0 after 300 epochs. This indicates that the model has converged and received effective training.

To provide a more intuitive display of the improved model's performance, a P-R curve was plotted for the improved model, as shown in Fig. 9. The horizontal axis represents the recall rate, while the vertical axis represents precision. The area under the curve (AUC) reflects the AP value. The larger the AP value, the closer the curve is to the coordinate (1,1), indicating a better model performance. As depicted in Fig. 9, the AP value of the "rappleyes" category is 96.9%, the "roccludedno" category achieves 92.7%, the "gappleyes" category attains 98.0%, and the "goccludedno" category reaches 90.9%, with a mAP of 94.6% for all categories. These results demonstrate that the model exhibits excellent detection performance.

In the natural orchard environment, fruits exhibit various shapes, and overlapping and occlusion phenomena are common due to foliage. To showcase the improved detection performance of the model, the test dataset images were used for predictions, yielding visual results of fruits under different natural conditions, as illustrated in Figs. 10 and 11. These figures demonstrate that the YOLOv5s-SDA algorithm effectively identifies both red and green apples, regardless of their quantity (single, 2–3, or multiple) or lighting conditions (front-lit, backlit, or shaded). The model also performs well in dense and occluded conditions, indicating its strong recognition capabilities towards fruits in complex environments after the improvements.

However, there are still cases of misclassification and missed detections, as shown in Fig. 12. In Fig. 12(a), the yellow-boxed apple on the left of the branch should be a non-grabbable red apple, but the model incorrectly identified it as a grabbable red apple. This misidentification could be due to the position of the branch at the edge of the fruit, where the classification features are not obvious, leading to a misclassification by the model. In Fig. 12 (b), the apple within the green circle is not recognized, possibly due to insufficient lighting resulting in a missed detection. In Fig. 12 (c), the apple within the yellow circle is incorrectly identified as a non-grabbable green apple, likely because the fruit is heavily occluded by leaves and positioned at the edge of the image. In Fig. 12 (d), two apples within the green circle are not detected, possibly because the fruits are heavily occluded by obstacles and have significant shadowing on their surfaces. In the future, optimizations can be made in the following areas: incorporating more practical scenarios to enhance the robustness of the model, attempting to adjust the network structure based on the characteristics of apple picking scenarios, and conducting systematic optimizations and adjustments.

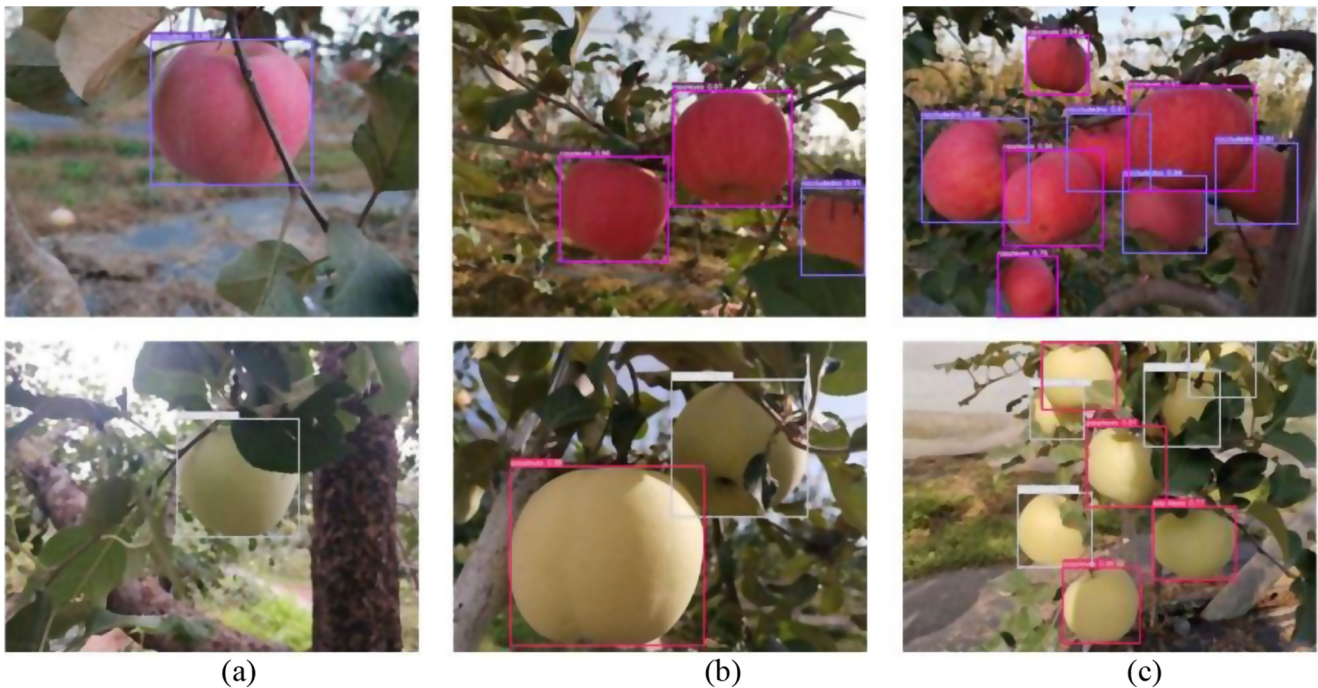


Fig. 10 Recognition performance of the model with varying apple quantities. (a) Single. (b) 2~3 pcs. (c) Multiple

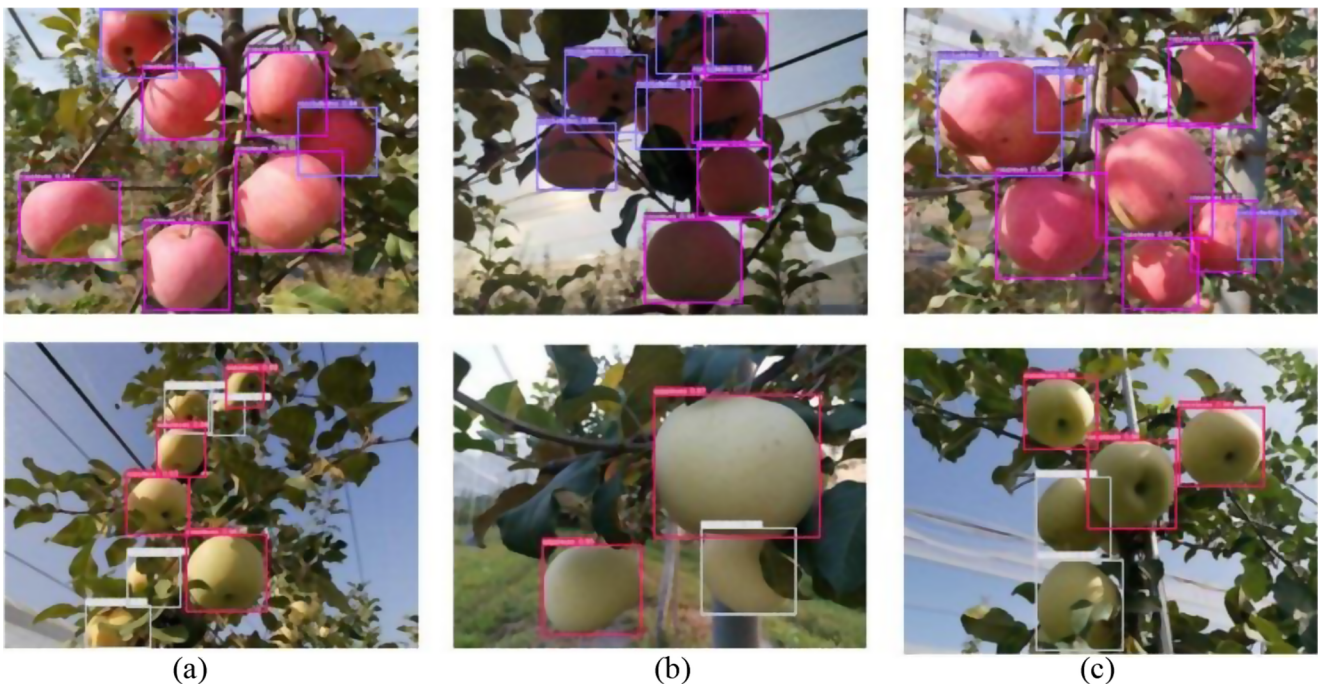


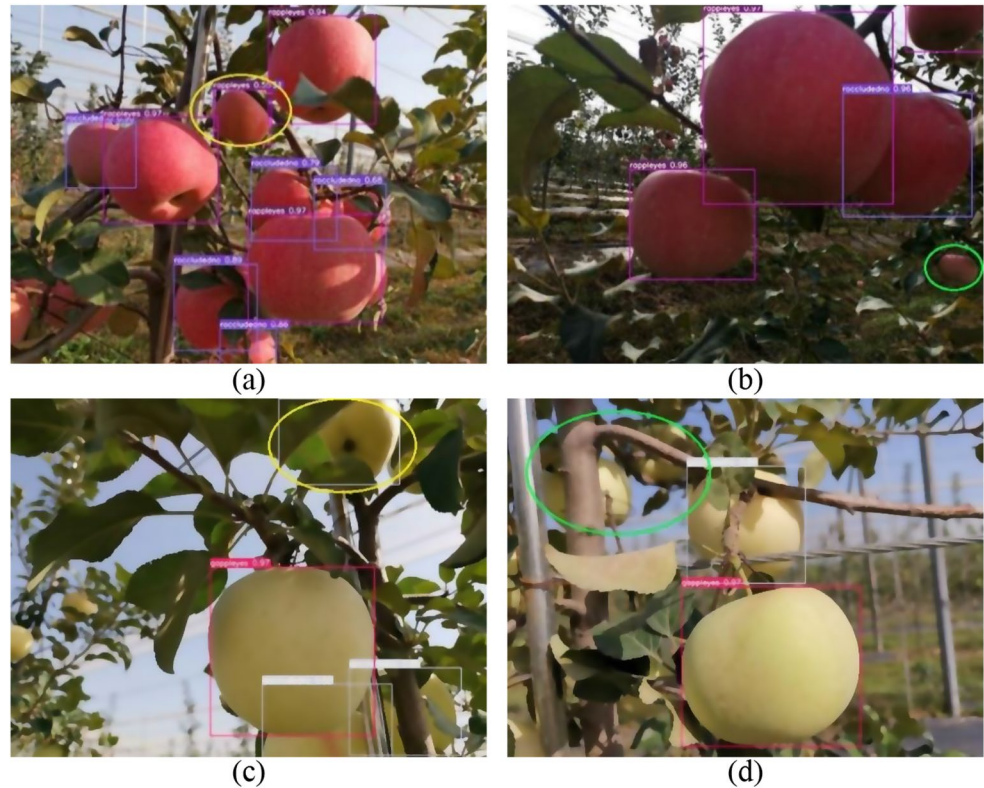
Fig. 11 Recognition results of the model under different lighting conditions. (a) Towards the light. (b) Backlight. (c) Shade

4 Conclusion

This study focuses on different growth conditions and colors of apples in natural environments, aiming to investigate the identification and detection of apples in situ using deep learning object detection algorithms. Furthermore, lightweight

enhancements were implemented on the YOLOv5s model to achieve apple recognition and localization in complex environments. To begin with, different datasets were collected for red and green apples in orchards under various conditions. To reduce mechanical damage to the branch and improve grabbing efficiency, the datasets were divided into

Fig. 12 Misclassification and missed detections of fruit. (a) The unpickable red apple error was misidentified as a pickable red apple (yellow circle). (b) Red apple missed detection (green circle). (c) The pickable green apple error was misidentified as an unpickable green apple (yellow circle). (d) Two heavily obscured fruits were not detected (green circle)



four categories: directly grabbable and non-directly grabbable red and green apples. These datasets were labeled using LabelImg software and split into training, validation, and testing sets following a 6:2:2 ratio. Secondly, various deep learning object detection algorithms were selected, including the two-stage algorithm Faster RCNN and the single-stage recognition algorithms YOLOv3, YOLOv3-Tiny, YOLOv5s, and YOLOv7. The YOLOv5s algorithm was found to have the best recognition and detection performance compared to others. The model was then enhanced with a lightweight backbone network, replacing the original backbone with different lightweight networks such as MobileNetV3, GhostConv, and ShuffleNetV2. After model prediction on the same testing set, the ShuffleNetV2 lightweight model was found to have the best performance. Therefore, further fine-tuning was performed to obtain the final lightweight model, YOLOv5s-SDA, with a mAP value of 94.6%, one order of magnitude fewer model parameters than the original YOLOv5s model, and the size of the weight file has been reduced from 14.4 MB to 1.7 MB, decreasing to 11.8% of its original size. The GFLOPs has been reduced from 16.4 to 2.5, with the floating-point operations representing only 15.2% of the original amount. Furthermore, the detection time per image was reduced from 20.7ms to 9.5ms, doubling the detection speed while maintaining model precision in natural environments. Overall, the YOLOv5s-SDA model showed excellent performance in fruit recognition,

with high accuracy and fast detection speed, and has great potential for future practical applications.

The models in this study were trained on computer devices and have not yet been tested in the field. As a result, there may be certain inaccuracies in their application to real-world environments. Subsequent research will focus on further refining the models to optimize their deployment on mobile devices. Moreover, further research will be conducted to enhance the accuracy of the model detection. This will involve the incorporation of additional data augmentation techniques and more robust target localization algorithms. Furthermore, exploration of additional application scenarios will be undertaken to maximize the effectiveness of fruit-picking robots in practical implementation.

Author contributions The first draft of the manuscript was written by LM and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Authors would like to acknowledge the financial support of the National Natural Science Foundation of China (Grant No. 31172064).

Data availability The datasets generated during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The author declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

- Tang, Y., Chen, M., Wang, C., Luo, L., Li, J., Lian, G., et al.: Recognition and localization methods for vision-based fruit picking robots: A review. *Front. Plant. Sci.* **11** (2020). <https://doi.org/10.3389/fpls.2020.00510>
- Peng, K., Ma, W., Lu, J., Tian, Z., Yang, Z.: Application of machine vision technology in citrus production. *Appl. Sci. (Basel)*. **13** (2023). <https://doi.org/10.3390/app13169334>
- Voulodimos, A., Doulamis, N., Doulamis, A., Protoukarakis, E.: Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.* **2018** (2018). <https://doi.org/10.1155/2018/7068349>
- Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., Lew, M.S.: Deep learning for visual understanding: A review. *Neurocomputing*. **187**, 27–48 (2016). <https://doi.org/10.1016/j.neucom.2015.09.116>
- Saleem, M.H., Potgieter, J., Arif, K.M.: Automation in agriculture by machine and deep learning techniques: A review of recent developments. *Precis Agric.* **22**, 2053–2091 (2021). <https://doi.org/10.1007/s11119-021-09806-x>
- Hasan, A.S.M.M., Sohel, F., Diepeveen, D., Laga, H., Jones, M.G.K.: A survey of deep learning techniques for weed detection from images. *Comput. Electron. Agric.* **184** (2021). <https://doi.org/10.1016/j.compag.2021.106067>
- Tang, Y., Qiu, J., Zhang, Y., Wu, D., Cao, Y., Zhao, K., et al.: Optimization strategies of fruit detection to overcome the challenge of unstructured background in field orchard environment: A review. *Precis Agric.* **24**, 1183–1219 (2023). <https://doi.org/10.1007/s11119-023-10009-9>
- Karthikeyan, M., Subashini, T.S., Srinivasan, R., et al.: YOLO-APPLE: Augment Yolov3 deep learning algorithm for apple fruit quality detection. *SIViP*. **18**, 119–128 (2024). <https://doi.org/10.1007/s11760-023-02710-z>
- Omer, S.M., Ghafoor, K.Z., Askar, S.K.: Lightweight improved yolov5 model for cucumber leaf disease and pest detection based on deep learning. *SIViP*. (2023). <https://doi.org/10.1007/s11760-023-02865-9>
- Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**, 1137–1149 (2017). <https://doi.org/10.1109/TPAMI.2016.2577031>
- Zhuang, X., Zhang, T.: Detection of sick broilers by digital image processing and deep learning. *Biosyst Eng.* **179**, 106–116 (2019). <https://doi.org/10.1016/j.biosystemseng.2019.01.003>
- 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.* **163** (2019). <https://doi.org/10.1016/j.compag.2019.06.001>
- Wan, S., Goudos, S.: Faster r-cnn for multi-class fruit detection using a robotic vision system. *Comput. Netw.* **168** (2020). <https://doi.org/10.1016/j.comnet.2019.107036>
- Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. 2016 IEEE Conf. Comput. Vis. PATTERN Recognit. (CVPR). **779–788** (2016). <https://doi.org/10.1109/CVPR.2016.91>
- Zhao, D.A., Wu, R.D., Liu, X.Y., Zhao, Y.Y.: Apple positioning based on YOLO deep convolutional neural network for picking robot in complex background. *Trans. Chin. Soc. Agric. Eng.* **35**, 164–173 (2019)
- Xuan, G., Gao, C., Shao, Y., Zhang, M., Wang, Y., Zhong, J., et al.: Apple detection in natural environment using deep learning algorithms. *IEEE Access.* **8**, 216772–216780 (2020). <https://doi.org/10.1109/ACCESS.2020.3040423>
- Peng, H.X., Huang, B., Shao, Y.Y., Li, Z.S., Zhang, C.W., Chen, Y., Xiong, J.T.: General improved SSD model for picking object recognition of multiple fruits in natural environment. *Trans. Chin. Soc. Agric. Eng.* **34**, 155–162 (2018)
- Chen, W., Zhang, J., Guo, B., Wei, Q., Zhu, Z.: An apple detection method based on des-yolo v4 algorithm for harvesting robots in complex environment. *Math. Probl. Eng.* **2021** (2021). <https://doi.org/10.1155/2021/7351470>
- Lv, J., Xu, H., Han, Y., Lu, W., Xu, L., Rong, H., et al.: A visual identification method for the apple growth forms in the orchard. *Comput. Electron. Agric.* **197** (2022). <https://doi.org/10.1016/j.compag.2022.106954>
- Zhu, L., Xie, Z., Luo, J., Qi, Y., Liu, L., Tao, W.: Dynamic object detection algorithm based on lightweight shared feature pyramid. *Remote Sens. (Basel)*. **13** (2021). <https://doi.org/10.3390/rs13224610>
- Wang, C., Wang, Z., Li, K., Gao, R., Yan, L.: Lightweight object detection model fused with feature pyramid. *Multimed Tools Appl.* **82**, 601–618 (2023). <https://doi.org/10.1007/s11042-022-12127-4>
- Wang, D., He, D.: Channel pruned Yolo v5s-based deep learning approach for rapid and accurate apple fruitlet detection before fruit thinning. *Biosyst Eng.* **210**, 271–281 (2021). <https://doi.org/10.1016/j.biosystemseng.2021.08.015>
- Ji, W., Pan, Y., Xu, B., Wang, J.: A real-time apple targets detection method for picking robot based on shufflenetv2-yolox. *Agric. (Basel)*. **12** (2022). <https://doi.org/10.3390/agriculture12060856>
- Xu, L., Wang, Y., Shi, X., Tang, Z., Chen, X., Wang, Y., et al.: Real-time and accurate detection of citrus in complex scenes based on hpl-yolov4. *Comput. Electron. Agric.* **205** (2023). <https://doi.org/10.1016/j.compag.2022.107590>
- Song, H.B., Shang, Y.Y., He, D.J.: Review on Deep Learning Technology for Fruit Target Recognition. *Trans. Chin. Soc. Agric. Mach.* **54**, 1–19 (2023)
- Wang, J., Feng, S.C., Cheng, Y.: Survey of research on lightweight neural network structures for deep learning. *Comput. Eng.* **47**, 1–13 (2021)
- Gao, F., Fu, L., Zhang, X., Majeed, Y., Li, R., Karkee, M., et al.: Multi-class fruit-on-plant detection for apple in snap system using faster r-cnn. *Comput. Electron. Agric.* **176** (2020). <https://doi.org/10.1016/j.compag.2020.105634>
- Yan, B., Fan, P., Wang, M.R., Shi, S.Q., Lei, X.Y., Yang, F.Z.: Real-time Apple picking Pattern Recognition for picking Robot based on improved YOLOv5m. *Trans. Chin. Soc. Agric. Mach.* **53**, 28–38 (2022)
- Yan, B., Fan, P., Lei, X., Liu, Z., Yang, F.: A real-time apple targets detection method for picking robot based on improved yolov5. *Remote Sens. (Basel)*. **13** (2021). <https://doi.org/10.3390/rs13091619>
- Hu, D., Zhu, J., Liu, J., Wang, J., Zhang, X.: Gesture recognition based on modified yolov5s. *IET Image Process.* **16**, 2124–2132 (2022). <https://doi.org/10.1049/ipr2.12477>
- Howard, A., Sandler, M., Chu, G., Chen, L., Chen, B., Tan, M., et al.: Searching for mobilenetv3. 2019 IEEE/CVF Int. Conf. Comput. Vis. (ICCV 2019). **1314–1324** (2019). <https://doi.org/10.1109/ICCV.2019.00140>
- Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C.: Ghostnet: More features from cheap operations. 2020 IEEE/CVF Conf. Comput. Vis. PATTERN Recognit. (CVPR). **1577–1586** (2020). <https://doi.org/10.1109/CVPR42600.2020.00165>

33. Zhang, X., Zhou, X., Lin, M., Sun, R.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. 2018 IEEE/CVF Conf. Comput. Vis. PATTERN Recognit. (CVPR). **6848–6856** (2018). <https://doi.org/10.1109/CVPR.2018.00716>
34. Ma, N., Zhang, X., Zheng, H., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. Comput. Vis. - ECCV 2018 PT XIV. **11218**, 122–138 (2018). https://doi.org/10.1007/978-3-030-01264-9_8
35. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. 30TH IEEE Conf. Comput. Vis. PATTERN Recognit. (CVPR 2017). **1800–1807** (2017). <https://doi.org/10.1109/CVPR.2017.195>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.