Detection of multi-class coconut clusters for robotic picking under occlusion conditions

Yuxing Fu[1,2], Hongcheng Zheng[2], Zongbin Wang[2], Jinyang Huang[2], Wei Fu[1,2*]

(1. *School of Information and Communication Engineering, Hainan University, Haikou 570228, China*;
2. *School of Mechanical and Electrical Engineering, Hainan University, Haikou 570228, China*)

Abstract: With the development of tree-climbing robots and robotic end-effectors, it is possible to develop automated coconut-picking robots with the help of machine vision technology. Coconuts grow in clusters in the canopy and are easily occluded by leaves. Therefore, the detection of multi-class coconut clusters according to the occlusion condition is necessary for robots to develop picking strategies. The coconut detection model, named YOLO-Coco, was developed based on the YOLOv7-tiny network. It detected coconuts in different conditions such as not-occluded, leaves-occluded, and trunk-occluded fruit. The developed model used Efficient Channel Attention (ECA) to enhance the feature weights extracted by the backbone network. Re-parameterization Convolution (RepConv) made the model convolution layers deeper and provided more semantic information for the detection head. Finally, the Bi-directional Feature Pyramid Network (BiFPN) was used to optimize the head network structure of YOLO-Coco to achieve the balanced fusion of multi-scale features. The results showed that the mean average precision (mAP) of YOLO-Coco for detecting multi-class coconut clusters was 93.6%, and the average precision (AP) of not-occluded, leaves-occluded, and trunk-occluded fruit were 90.5%, 93.8%, and 96.4%, respectively. The detection accuracy of YOLO-Coco for yellow coconuts was 5.1% higher than that for green coconuts. Compared with seven mainstream deep learning networks, YOLO-Coco achieved the highest detection accuracy in detecting multi-class coconut clusters, while maintaining advantages in detection speed and model size. The developed model can accurately detect coconuts in complex canopy environments, providing technical support for the visual system of coconut-picking robots.

Keywords: coconut clusters, picking robot, leaves-occluded, multi-class detection, YOLOv7-tiny
DOI: 10.25165/j.ijabe.20251801.9031

Citation: Fu Y X, Zheng H C, Wang Z B, Huang J Y, Fu W. Detection of multi-class coconut clusters for robotic picking under occlusion conditions. Int J Agric & Biol Eng, 2025; 18(1): 267–278.

1 Introduction

Cocos nucifera, commonly known as coconut, is a very important tree in the tropics. It provides food, employment, and business opportunities to millions of people, making significant contributions to economic and livelihood development in tropical regions of the world[1]. Hainan Province is the largest tropical province in China, with coconut cultivation and production accounting for over 90% of China's total, making it the main producer of coconuts in China[2]. As reported by the Hainan Provincial Bureau of Statistics, in 2022, the coconut cultivation area in Hainan was about 37 900 hm^2 and the annual production exceeded 220 million[3]. Coconut is rich in nutrients, and products produced from coconut as raw material involve multiple fields, such as food, chemicals, medicine, navigation, etc., which have become an indispensable part of daily life and many industries[4,5]. Despite the strong demand for coconut, because of its tall trunk and complex canopy structure, people engaged in coconut picking are required to have a rich tree-climbing experience and abundant physical strength. This increases the risk of manual harvesting of coconuts, and managers often face high costs.

With the rapid development of artificial intelligence and robotics, vision-based autonomous harvesting robots have become the preferred method for the selective picking of various fruits[6]. The picking robot uses a visual system to detect and locate fruits, and then an end-effector is used to separate fruits[7]. Over the past decade, coconut-picking robots have been developed with an end-effector of a circular, sharp blade that rotates at high speed for cutting the coconut stalk[8,9]. However, the coconut canopy is a complex environment where coconuts are easily occluded by petioles. If the robot attempts to pick occluded coconut clusters, the high-speed rotating end-effector may cause damage to the canopy and coconuts. The effectiveness of the end-effector operation largely depends on the performance of the fruit detection algorithm. Parvathi and Selvi[10] attempted to separate coconuts from the canopy using color-based segmentation, edge detection, and circular Hough transform, respectively. However, the similarity in color between coconuts and surrounding leaves and the overlap of coconuts with each other resulted in poor segmentation results. Traditional object detection algorithms cannot effectively detect all coconuts. Deep learning-based target recognition technology has the advantages of self-learning of target features, robustness to target occlusion and light changes, etc., and has been widely researched in agricultural fields such as fruit detection[11,12], fruit maturity grading[13], precision agriculture[14,15], leaf pests and diseases detection[16], and animal behavior detection[17].

In recent years, the application of deep learning in agriculture

Received date: 2024-04-27 **Accepted date:** 2024-12-12
Biographies: Yuxing Fu, PhD, research interest: agricultural informatization and detection technology, Email: fyx_aca@163.com; **Hongcheng Zheng**, Master candidate, research interest: agriculture robot, Email: zhchainanu@163.com; **ZongbinWang**, Master candidate, research interest: intelligent equipment and technology, Email: 923504289@qq.com; **Jinyang Huang**, Master candidate, research interest: agricultural informatization, Email: 1127314510@qq.com.
***Corresponding author: Wei Fu**, PhD, Professor, research interest: intelligent agricultural machinery equipment. School of Mechanical and Electrical Engineering, Hainan University, Haikou 570228, China. Tel: +86-18608973300, Email: 994026@hainanu.edu.cn.

has grown significantly, showing great potential in the field of fruit detection such as apples, citrus, kiwifruit, grape, and dragon fruit[18-20]. Zhang et al.[21] used VGG19 to improve Faster R-CNN, which effectively improves the accuracy of apple detection. In order to avoid robots forcibly picking apples occluded by branches or wires, Gao et al.[22] proposed a Faster R-CNN-based method for detecting multi-class apples in dense-leafed fruit trees. The study had a mAP of 87.9% and processed each image with a resolution of 1920×1080 at an average speed of 241 ms. Two-stage object detection separates target localization and target classification, that is, first generating candidate regions, and then classifying them. Although it has high detection accuracy, the number of parameters and calculation amount are large, and the detection speed is slow. The choice of target detection model needs to take into account both accuracy and speed. One-stage object detection directly generates the class probability and position coordinates of the target, which achieves fast detection with high accuracy[23]. Thus, it is widely used in fruit detection tasks. In order to detect four kinds of fruits, including apple and lychee, Peng et al.[24] replaced the original backbone network of SSD with ResNet-101, and optimized the model with a transfer learning method and stochastic gradient descent algorithm. You Only Look Once (YOLO) is an object detection algorithm proposed by Redmon[25]. YOLO has fast detection speed with high accuracy and excellent generalization performance. Li et al.[26] proposed a tomato detection and localization algorithm based on improved YOLOv5s to meet the requirements of an intelligent tomato-picking process. The mAP of the algorithm was up to 99.77%, and the average detection speed per image was 9 ms. Xu et al.[27] adopted lightweight feature network GhostNet as the backbone network of YOLOv4 to enhance the feature extraction of citrus, improving the detection speed of citrus in complex scenes while ensuring detection accuracy. The model size was also reduced for easy deployment to embedded devices. The above studies classified fruits in different occlusion conditions into one class and did not fully consider the fruit detection needs in the robot-picking scene. However, Suo et al.[28] and Zhang et al.[29], respectively, proposed detection methods for multi-class kiwifruits and cherry tomatoes based on YOLOv4. The fruits were detected and labeled into four classes: fruit not occluded, fruit occluded by leaves, fruit occluded by other fruits, and fruit occluded by branches. The method avoided fruits that were occluded by branches or other fruits as picking targets. From the perspective of robot picking, their study avoids potential damage to robots and fruit trees and has important practical significance for guiding robot picking.

Accurate detection of coconut clusters based on their occlusion conditions is necessary for planning the motion of the robotic end-effector. There have been no previous reports on using YOLO as the main method for coconut detection. Therefore, this study used the advanced YOLOv7-tiny lightweight network to develop the detection model of multi-class coconut clusters. To further improve detection performance, this study conducted the following: 1) The ultra-light attention module Efficient Channel Attention (ECA) was introduced to enhance the feature weight of the backbone network input to the head network and weaken the background information; 2) RepConv was introduced to provide rich feature information for the detection head; and 3) BiFPN was used to improve the head network to achieve a weighted bidirectional fusion of features at different scales. The detection model developed in this study will provide technical support for object perception for coconut-picking robots.

2 Materials and methods

2.1 Image acquisition

Coconut images were obtained from a commercial coconut orchard in Dongjiao Town, Wenchang City, Hainan Province, China, as shown in Figure 1. Green and yellow coconuts were planted in the orchard. At that time, coconuts were grown for 6-7 months and were mainly used as fruit drinks. The Nikon D90 camera was used for image acquisition, and the image resolution was 4288×2848. In this study, the robot's perspective was simulated to obtain images, due to the camera of the coconut-picking robot's visual system facing upwards towards the canopy. Users stood at multiple locations around the trunk to obtain coconut images so that different parts of the canopy were covered. Images were obtained at different times of the day, and 1344 images with clear object contours and textures were selected as the dataset.

a. Coconut orchard scene b. Yellow coconuts c. Green coconuts

Figure 1 Coconut cluster images from a commercial orchard

2.2 Coconut cluster dataset

2.2.1 Classification criteria

Coconut leaves include petiole, rachis, and leaflets, as shown in Figure 2. The petiole plays an important role in supporting the weight of coconut clusters and preventing them from falling to the ground. However, this causes coconut clusters to be occluded, which is not conducive to robot picking. Thus, according to the occlusion conditions in the canopy, coconut clusters are divided into three classes.

The first class indicates that the coconut cluster is not occluded (referred to as NO in this study), as shown in Figure 3a. The second class indicates that the coconut cluster is occluded by leaves (referred to as OL), and the coconut cluster is mainly occluded by the petiole, as shown in Figure 3b. NO and OL represent coconut clusters located in the picking area. The third class indicates that the coconut cluster is occluded by the trunk (referred to as OT), as shown in Figure 3c. OT is the coconut cluster located at the back of the picking area. In addition, if the coconut cluster is occluded by both leaves and trunk, it is classified as OT.

Figure 2 Occluded coconuts and structure of coconut leaf

2.2.2 Dataset labeling and partitioning

Before training, LabelImg was used to manually label the coconut cluster in the image, in which fruit not occluded was labeled as NO, fruit occluded by leaves was labeled as OL, and fruit occluded by trunk was labeled as OT. During labeling, the rectangular box should fit the outline of coconut clusters as much as possible. If the object is occluded by leaves or trunk, the user draws the minimum bounding rectangle of the occluded coconut cluster based on personal experience to depict the actual size of the object. Due to the fact that coconuts can bloom and fruit throughout the year, a few young fruits are present in the image. To avoid the interference of small object fruits, they are left as part of the background and are not labeled. Once the labeling is complete, a text file containing object class and coordinate information is generated.

The original test set consists of A+B, with a total of 244 images. Test set A is a yellow coconut image, and test set B is a green coconut image. Then, 978 images were randomly selected from the remaining original images as the training set and 122 as the validation set. In order to improve the generalization ability of the model, image rotation, brightness adjustment, and mirroring (Figure 4) were used to expand the original training set, verification set, and test set to 3912, 488, and 976 images, respectively. After data augmentation, dataset distribution is shown in Figure 5.

a. Not occluded (NO) b. Occluded by leaves (OL) c. Occluded by trunk (OT)

Figure 3 Schematic diagram of three classes of coconut clusters and their labeling

a. Original image b. Rotated image c. Brightness-adjusted image d. Mirrored image

Figure 4 Image data after data enhancement

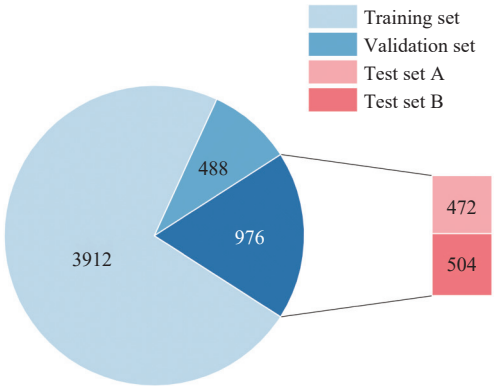


Figure 5 Quantity distribution of datasets

2.3 Model building

YOLOv7-tiny is a lightweight model of the YOLOv7 series for edge computing devices, which achieves high accuracy and speed on the publicly available Microsoft COCO Dataset[30]. The complete network structure of YOLOv7-tiny can be divided into three parts: Input, Backbone, and Head[31].

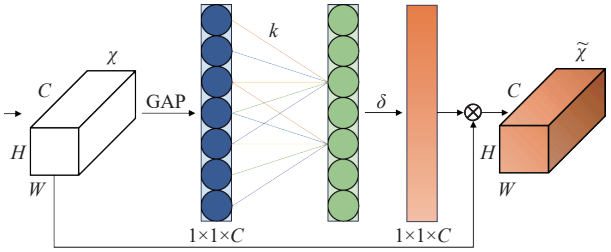
Input mainly preprocesses the input data and uses a dynamic label allocation strategy to determine the positive and negative samples. Input mainly preprocesses the training samples and utilizes Mosaic for online data augmentation. The backbone consists of a convolutional module, Efficient Layer Aggregation Network (ELAN), and Max Pooling (MP). The convolutional module consists of a convolutional layer, a Batch Normalization (BN) layer, and an activation function. ELAN controls the shortest and longest gradient paths and continuously enhances the feature learning ability of the model through a deeper network. MP achieves the final down-sampling operation by fusing two down-sampling branches. SPPCSP consists of Spatial Pyramid Pooling (SPP) and Cross-Stage Partial (CSP) in the head network. CSP divides the features into two parts. One part obtains four different scale receptive fields through the Max Pooling operation in SPP, and the other part performs the convolution operation. The results of the two parts are finally fused to enrich the feature information. Detect head outputs the coordinates, confidence, and class information of the object bounding box.

2.3.1 Efficient channel attention mechanism

ECANet proposed a local cross-channel exchange strategy without dimensionality reduction based on the SE module[32]. The strategy allows a direct link between channels and weights. Appropriate cross-channel interactions both maintain performance and significantly reduce model complexity. The module can achieve a significant performance improvement with only a few additional parameters. As can be seen from Figure 6, ECA first compresses the feature map by inputting the height of the feature map (H) × the width of the feature map (W) × the number of channels (C), i.e., transforms it into a feature map of size 1×1×C by global average pooling (GAP). Then, ECA performs one-dimensional convolution on the 1×1×C feature map through a one-dimensional convolution kernel of size k to extract the relationship between k channels and complete the cross-channel information exchange. The parameter k, which is proportional to the number of channels and can adapt to the channel dimension of the input, is defined as follows:

$$k = |x|_{odd} = \left| \frac{\log_2 C + 1}{2} \right|_{odd} \quad (1)$$

where, $|x|_{odd}$ represents the odd number closest to x, and C is the total number of channels. ECA generates a weight ratio for each feature channel using an activation function. Subsequently, ECA combines the original H×W×C input features with the channel weights. Thus, the important features in the feature map are given large weights, and the useless features are given small weights, to improve the ability of feature representation.



δ: Sigmoid activation function ⊗: Element-wise product

Notes: H, W, and C represent the height, width, and number of channels of the feature map, respectively. χ and $\tilde{\chi}$ are input and output, respectively. GAP and k are global average pooling and the number of convolution kernels, respectively.

Figure 6 Schematic diagram of the ECA mechanism

2.3.2 Re-parameterization convolution block

Coconut clusters grow in complex environments and are heavily occluded by leaves. YOLOv7-tiny has fewer convolutional layers than other models in the v7 series. Deeper convolutional layers help to learn object features[33]. In order to improve the learning ability of coconut cluster features, RepConv is introduced near the detection head. RepConv can construct a multi-branch topology in the training phase to extract more effective semantic features. Then, the re-parameterization technique can transform the multi-branch structure into a single-branch structure for inference, simplifying the model and speeding up the inference[34].

It can be seen that the RepConv block training model is composed of three branches in Figure 7a. The first layer is a 3×3 convolution for feature extraction. The second layer is a 1×1 convolution for smoothing features. The third layer is the identity branch, which only performs BN operations on the input features. Finally, three branches are added, and the final result is output by the activation function, thus enhancing the model expression ability. In the re-parameterization phase, the BN layer and its preceding convolutional layer in each branch were converted into a convolutional layer with bias. Let $\{W_i', b_i'\}$ be the i-branch converted convolution kernel and bias, which can be obtained as follows:

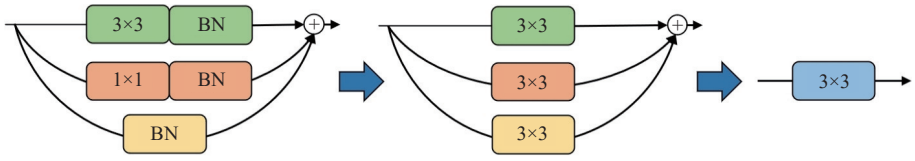
$$\mathrm{Conv}\left(W^{(i)}\right) = M * W^{(i)} \quad (2)$$

$$\mathrm{BN}\left(\mathrm{Conv}\left(W^{(i)}\right)\right) = \gamma^{(i)} \frac{\mathrm{Conv}\left(W^{(i)}\right) - \mu^{(i)}}{\sigma^{(i)}} + \beta^{(i)} \quad (3)$$

$$W_i' = \frac{\gamma^{(i)}}{\sigma^{(i)}} W^{(i)} \quad (4)$$

$$b_i' = \beta^{(i)} - \frac{\mu^{(i)}\gamma^{(i)}}{\sigma^{(i)}} \quad (5)$$

where, M is the input; $W^{(i)}$ is the i-branch convolution kernel; and $\mu^{(i)}$, $\sigma^{(i)}$, $\gamma^{(i)}$, and $\beta^{(i)}$ are the accumulated mean, standard deviation, learned scaling factor, and bias of the i-branch BN layer following 3×3 convolutional layers, respectively. This transformation method is also applicable to the identity branch since it can be considered as a 1×1 convolution with the unit matrix as kernel. After such transformations, a 3×3 convolution kernel, two 1×1 convolution kernels, and three biases were obtained, and the two 1×1 kernels will be padded by zeros to 3×3, as shown in Figure 7b. According to the linear property of convolution, the final single 3×3 convolution kernel and bias are obtained by adding three kernels and three biases, respectively, as shown in Figure 7c. The re-parameterization technique involves three main transformations, i.e., fusion of the BN layer and its preceding convolutional layer, BN layer transformation of the identity branch, and addition of three branches. The re-parameterized inference model output is the same as the multi-branch output.



a. RepConv training b. A conv layer converted by BN and its preceding conv layer c. RepConv inference

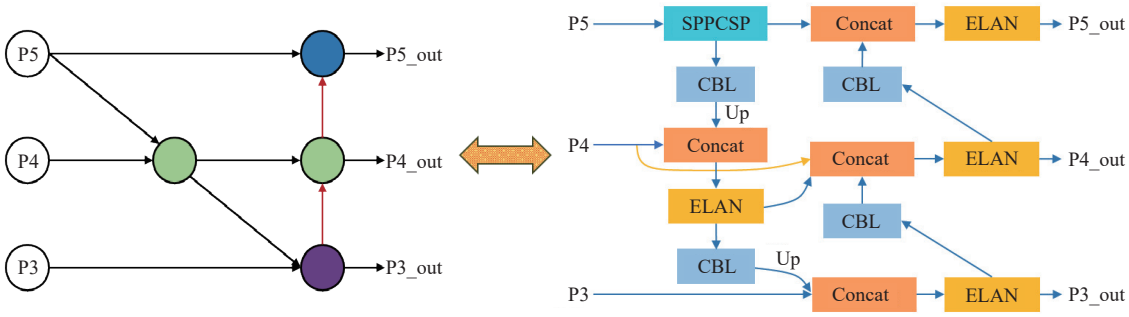
Note: BN: Batch Normalization. Same below.

Figure 7 Re-parameterization process

2.3.3 Bi-directional feature pyramid network

Path Aggregation Network (PANet) fuses different input features using two paths, bottom-up and top-down, to enhance the representational capability of the backbone network and achieve high-level feature fusion. These input features have different resolutions, and their contributions to the fused output features are not equal, which can result in a few features being ignored. Therefore, BiFPN is used to improve PANet[35]. BiFPN learns the

importance of different input features by introducing learnable weights and bidirectional connections to achieve cross-level fusion of multi-scale features with weights. In addition, BiFPN fuses the original features extracted from the backbone network with high-level semantic features by adding a path between the original input node and the output node at the same layer. This solves the problem of missing a few features and is conducive to extracting richer features. The BiFPN structure of the 3-layer structure is shown in Figure 8a. Feature P5 is not output directly, but aggregates features P3 and P4 from top to bottom through up-sampling (Up), and then P3_out is obtained. Feature P4 aggregates up-sampled P5 and scale-compressed P3_out to obtain P4_out. P5_out is obtained by the aggregation of feature P5 and scale-compressed P4_out. BiFPN's improved head network is shown in Figure 8b.



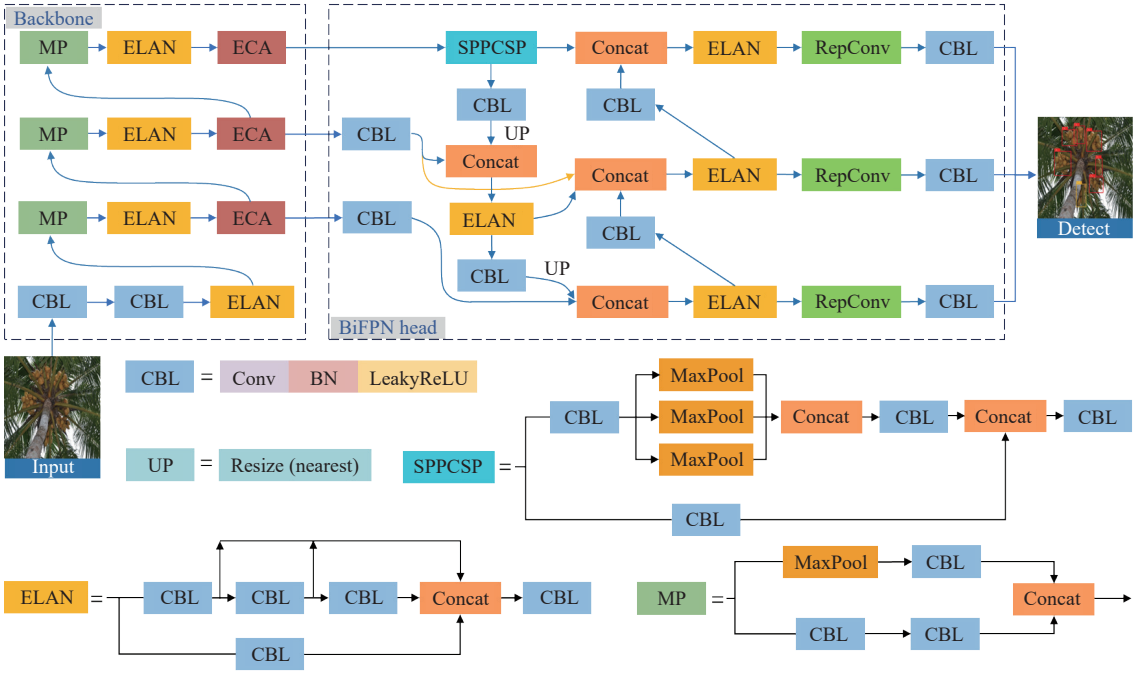
a. BiFPN with 3-layer structure b. Improved head network with BiFPN

Notes: SPPCSP: Consists of Spatial Pyramid Pooling (SPP) and Cross-Stage Partial (CSP); Concat: Concatenate; ELAN: Efficient Layer Aggregation Network; CBL: Convolution+BN+LeakyReLU; Up: upsampling. Same below.

Figure 8 Network structure diagram of BiFPN and improved head network with BiFPN

2.3.4 Construction of YOLO-Coco

Figure 9 shows the structure of YOLO-Coco improved using ECA, RepConv, and BiFPN. ECA is added at three positions where the backbone features enter the head network[36], because ECA can enhance coconut cluster features entering the head network and suppress background expression through cross-channel information exchange. To enhance the semantic richness detected by the detection head, RepConv is added in front of the CBL layer close to the detection head. Finally, BiFPN is introduced in the head network to achieve a differentiated fusion of features at different layers by learning the importance of different input features, which further improves the model detection accuracy. YOLO-Coco has a multi-scale and multi-branch structure similar to the Inception module during training, which will greatly improve the learning ability of the network. When inference is performed, the model converts from a multi-branch structure to a single-branch structure, which reduces the computation and memory consumption and achieves the improvement of inference speed.



Notes: MP: Max Pooling neuron; ECA: Efficient Channel Attention; Conv: Convolution.

Figure 9 YOLO-Coco network structure diagram

2.4 Model training and performance evaluation

The parameters of the experimental platform are listed in Table 1. The input image size received by YOLO-Coco is 640×640 and a stochastic gradient descent optimizer was used. The network training hyperparameters are listed in Table 2.

After model training, the AP, mAP, F1-score, and frames per

second (FPS) were used to evaluate model performance. The AP_i is calculated by Precision (P_i) and Recall (R_i), and P_i and R_i are defined as follows:

$$R_i = \frac{TP_i}{TP_i + FN_i} \tag{6}$$

$$P_i = \frac{TP_i}{TP_i + FP_i} \tag{7}$$

where, i represents the ith class: NO (i = 1), OL (i = 2), OT (i = 3). TP_i is the number of correctly detected coconut clusters, FP_i is the number of falsely detected coconut clusters, and FN_i is the number of non-detected coconut clusters. F1-score is defined as the harmonic average of P and R, as shown in Equation (8). AP_i is defined in Equation (9) as the area under P_i-R_i curves, which is an important indicator for measuring the performance of the object detection model. mAP, which is the average AP of the three classes, is shown in Equation (10).

$$\text{F1} - \text{Score} = 2 \cdot \frac{P \cdot R}{P + R} \tag{8}$$

$$AP_i = \int_0^1 P_i(R_i)\mathrm{d}R_i \tag{9}$$

$$\text{mAP} = \frac{1}{k}\sum_{i=1}^{k} AP_i \tag{10}$$

Table 1 Experimental platform setup

Items	Parameters
CPU	Intel(R) Core(TM) i7-12900 K
GPU	NVIDIA GEFORCE RTX 3060
Operating system	Windows 10
Acceleration environment	CUDA11.6, CUDNN8.3.2
Development platform	Python3.10, Pytorch 1.13.1

Table 2 Hyperparameters in the training process

Hyperparameter	Value
Epoch	300
Batch size	16
Momentum	0.937
Initial learning rate	0.001
Weight decay	0.0005

3 Results and analysis

3.1 Experiments on the changes in the detection model

To verify the effectiveness of different improvements in the proposed model, a performance comparison was conducted on different improved structures using a complete test set, and the results are shown in Table 3. It can be seen that ECA uses channel weights to calibrate features to achieve coconut feature enhancement and natural background suppression, and the mAP of the model increases by 1.2%. But the p-value slightly decreased. Then, RepConv is added before the last convolutional layer of each of the three detection branches. RepConv improves the learning ability of the network through the multi-branch structure, increasing the mAP to 92.7%. RepConv uses re-parameterization to improve the inference speed, but the model has deeper layers than the original model, and the number of parameters and floating point operations (FLOPs) are increased by 0.35 M and 0.7 G, respectively. RepConv is the main reason for the increase in the number of parameters and FLOPs of the model. BiFPN can effectively improve the problem of missing a few important features in PANet by learning the importance of input features with different resolutions. Compared with the original model, although the parameters of YOLO-Coco are increased by 0.36 M, P, R, and mAP are increased by 1.5%, 5.8%, and 3.0%, respectively. This means that YOLO-Coco has higher accuracy in predicting coconut clusters. However, the synergy of ECA, RepConv, and BiFPN effectively improved the detection performance of the model, increasing the mAP of the model from 90.6% to 93.6%.

Table 3 Experimental results of the detection model changes

Model	① ECA	② RepConv	③ BiFPN	P/%	R/%	mAP/%	F1-score	Parameters	FLOPs/G
YOLOv7-tiny				88.1	82.5	90.6	0.852	6 013 008	13.0
YOLOv7-tiny+①	√			86.1	85.4	91.8	0.857	6 013 011	13.0
YOLOv7-tiny+①+②	√	√		87.4	87.7	92.7	0.876	6 357 971	13.7
YOLOv7-tiny+①+②+③ (**YOLO-Coco**)	√	√	√	89.6	88.3	93.6	0.889	6 374 364	13.7

Notes: P: Precision; R: Recall; mAP: Mean Average Precision.

3.2 Comparison of different attention mechanisms

To assess the performance of ECA attention mechanisms, Squeeze-and-Excitation (SE), Convolutional Block Attention Module (CBAM), Coordinate Attention (CA), and Simple Attention Module (SimAM) were used to compare the detection performance. These modules are highly valued in agriculture. In the context of this study, the same number of attention modules were added at the same locations of the model. Among the five types of attention modules, ECA has the most significant improvement in the detection performance of the model, with mAP reaching 93.6%, as listed in Table 4. The mAP of the ECA model exceeded the SE module by 1.8%, the CBAM module by 1.4%, the CA module by 1.7%, and the SimAM module by 0.8%. Meanwhile, the model using ECA has the highest p-value. This ensures that the model can more accurately predict objects in complex agricultural environments. In addition, ECA, SE, CA, and SimAM are nearly identical in terms of the number of parameters, FLOPs, and size. Although CBAM has a mAP of 92.2%, it has higher parameters, FLOPs, and size. Overall, compared to the other four attention modules, the model improved using ECA and has higher accuracy while maintaining lower FLOPs and size.

Table 4 Detection results under different attention mechanisms

Model	Number	P/%	R/%	mAP/%	Parameters	FlOPs/G	Size/MB
X+ECA	3	89.6	88.3	93.6	6 374 364	13.7	12.7
X+SE	3	88.0	83.4	91.8	6 418 321	13.8	12.8
X+CBAM	3	87.2	88.1	92.2	6 763 519	14.4	13.5
X+CA	3	86.7	86.0	91.9	6 410 041	13.8	12.8
X+SimAM	3	88.4	88.0	92.8	6 374 361	13.7	12.7

Note: X is the part of the improved model after removing the attention module.

3.3 Visual analysis of the detection of different classes of coconut clusters

Figure 10 shows the P-R curves of YOLO-Coco and its original model for detecting each class. The APs of NO, OL, and OT in YOLO-Coco increased by 1.7%, 1.1%, and 6.0%, respectively. To explore the regions of interest that class features focus on, the detection results of the model are visualized using heatmaps. The region brightness is used to indicate its share in the prediction output process, with brighter colors indicating more attention. Figure 11 visually shows the difference in the focus on class

features of coconut clusters between YOLO-Coco and the original model. NO focuses on the central position of coconut clusters and radiates the boundaries of coconut clusters. OL takes both coconut clusters and leaves as the judgment criteria, which also causes the model to pay some attention to the area with only leaves. The region of interest of OT spreads from the center of the coconut clusters to the trunk. Compared with the OL class, OT has simpler regions of interest, and it is easier for the target to obtain higher confidence results. Compared with the visualization results of different models, it is found that YOLO-Coco focuses on more precise class feature regions, has stronger anti-interference ability, and has a lower probability of false positives.

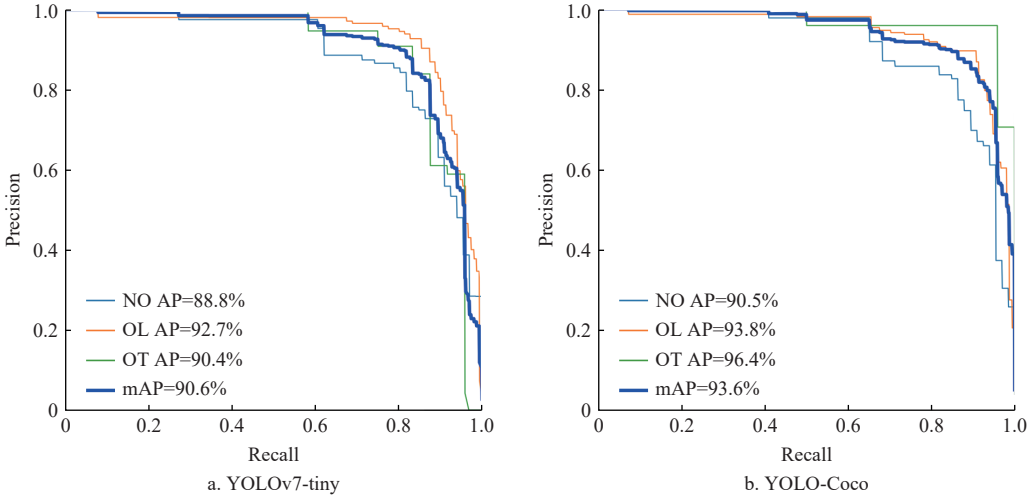
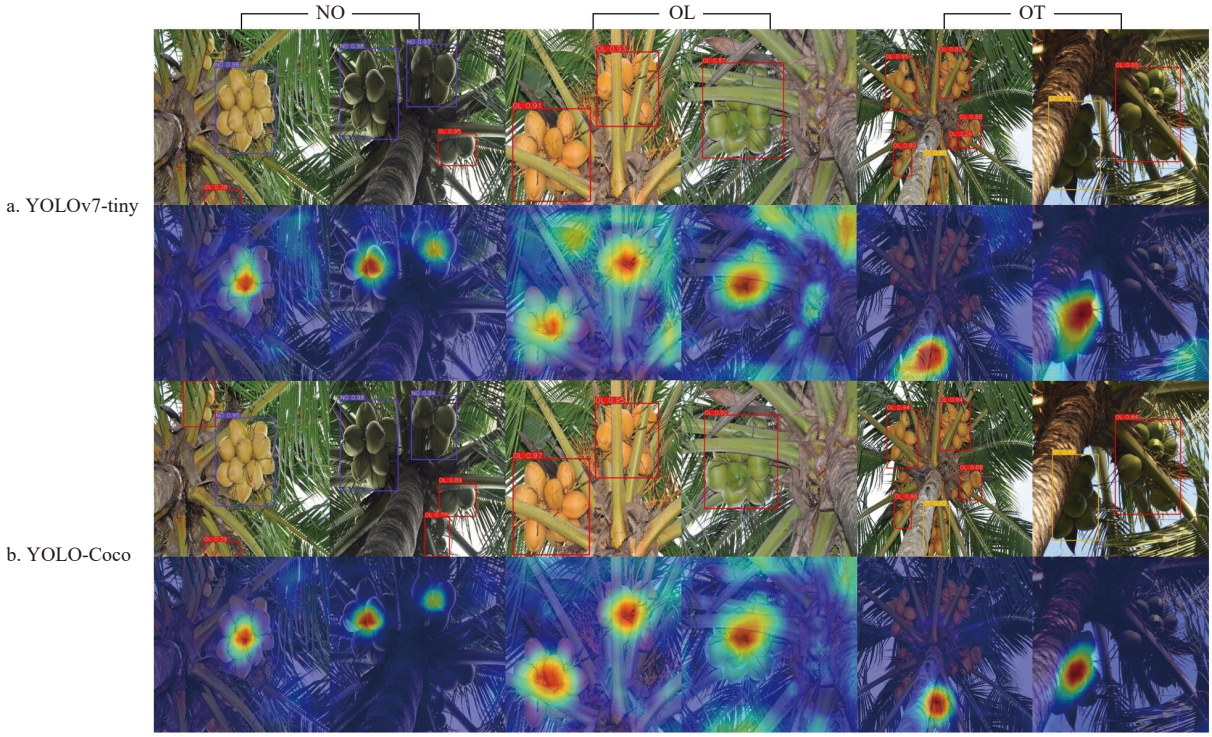


Figure 10 *P-R* curves of each object detected by YOLOv7-tiny and YOLO-Coco



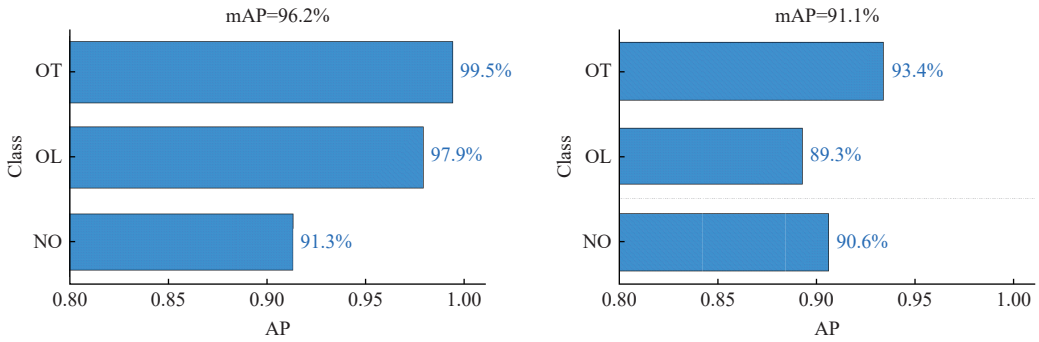
Note: NO, OL, and OT respectively refer to the class features explored in images.

Figure 11 Visualization of regions of interest for each class feature detected

3.4 Detection performance on different varieties of coconuts

To evaluate the performance of YOLO-Coco on yellow coconuts and green coconuts, test A and test B were used to test the model performance respectively, and the results are shown in Figure 12. The mAP of yellow coconuts was 96.2%, and the mAP of green coconuts was 91.1%. The model has shown good results in the detection of both varieties of coconuts. The color difference between yellow coconuts and the coconut canopy is significant, so these coconuts are easier to detect. The surface color of the green coconuts tends to be consistent with that of the leaves. Therefore, the detection accuracy of the three classes of yellow coconuts is higher than that of green coconuts. Since the fruit is not occluded, NO can present complete coconut cluster characteristics, and there is little difference in AP between the two varieties of coconuts. OL green coconuts are similar in color to the leaves and showed poor AP, but AP of OL yellow coconuts is as high as 97.9%, which is 8.6% higher than that of OL green coconuts. This indicates that in the OL class, YOLO-Coco not only focuses on the shape and texture of the coconut but also pays more attention to the color characteristics of the coconut. Even when coconut clusters are

occluded by the trunk, there is a significant difference in color between the trunk and the two varieties of coconuts, and the class characteristic is relatively simple. Compared to the other two classes, OT has simpler class features, so its AP is the highest among each coconut variety. The detection results of the two varieties of coconuts show that the detection performance of YOLO-Coco in yellow coconuts is better than that in green coconuts.



a. Yellow coconut clusters b. Green coconut clusters

Figure 12 Detection results of YOLO-Coco on different varieties of coconuts

3.5 Comparison with other deep learning detection models

The purpose of developing YOLO-Coco is to improve detection accuracy with a high detection speed. At present, deep learning algorithms have been widely applied in the field of fruit object detection. To further evaluate the performance advantages of YOLO-Coco, seven mainstream deep learning models were selected under the same training platform configuration and dataset for the detection of multi-class coconut clusters.

3.5.1 Performance comparison of mainstream detection networks

The detection results of multi-class coconut clusters using different mainstream deep learning networks are listed in Table 5. Compared with Faster R-CNN, YOLOv3, YOLOv4, and YOLOv7, YOLO-Coco has faster detection speed and smaller model size and has significant advantages in deploying edge computing devices. YOLOv3-tiny, YOLOv5s, and YOLOv8n are all lightweight network models. Compared with YOLOv3-tiny, the mAP of YOLO-Coco has increased by 4.9%, and the AP of NO, OL, and OT coconut clusters have all increased by more than 3%. YOLO-Coco has better attention to NO and OT coconut clusters. YOLOv5s has a higher detection speed, but its mAP and P are lower than YOLO-Coco. AP of NO and OL detected by YOLOv5s were closer to YOLO-Coco, but AP of OT was much lower than that of YOLO-Coco. YOLOv8n is the latest YOLO deep learning model with the fastest detection speed. Its model size is about 50% smaller than YOLO-Coco, and its detection speed is 217.4 FPS. The detection accuracy of NO detected by YOLOv8n is 1.1% higher, but in terms of OL and OT, the detection accuracy of YOLO-Coco is 3.9% and 6.8% higher, respectively. In addition, in coconut detection, the FPS of Faster R-CNN is only 4.8, which is significantly lower than that of the one-stage YOLO object detection model, indicating the limitations of the two-stage detection model in real-time detection.

Table 5 Results of different deep learning models for detecting multi-class coconut clusters

Model	*P*/%	*R*/%	mAP/%	Class AP/%			Detection speed/fps	Size/MB
				NO	OL	OT		
Faster R-CNN	77.8	79.1	80.5	84.5	88.4	68.6	4.8	115.3
YOLOv3-tiny	75.9	87.6	88.7	84.3	90.2	91.6	149.3	16.6
YOLOv3	86.6	82.1	88.3	88.2	86.8	90.0	33.3	117.8
YOLOv4	79.2	81.7	85.9	84.5	92.0	81.3	59.5	100.6
YOLOv5s	85.4	88.2	90.5	87.1	92.6	91.9	166.7	13.9
YOLOv7	86.2	83.5	91.0	88.7	**93.8**	90.6	65.4	71.3
YOLOv8n	81.9	87.1	90.4	**91.6**	89.9	89.6	**217.4**	**6.0**
YOLO-Coco	**89.6**	**88.3**	**93.6**	90.5	**93.8**	**96.4**	163.9	12.7

Note: The best results are highlighted in bold.

As can be seen from Figure 13, the AP of each class detected by YOLO-Coco ranks first in OT, ranks first with YOLOv7 in OL, and ranks second in NO, only behind the latest YOLOv8n. YOLO-Coco has demonstrated high accuracy with a high speed in the detection of multi-class coconut clusters. Furthermore, its lightweight model size makes it easier to meet platform portability and deployment requirements.

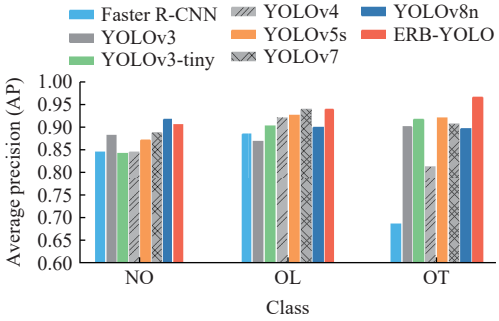
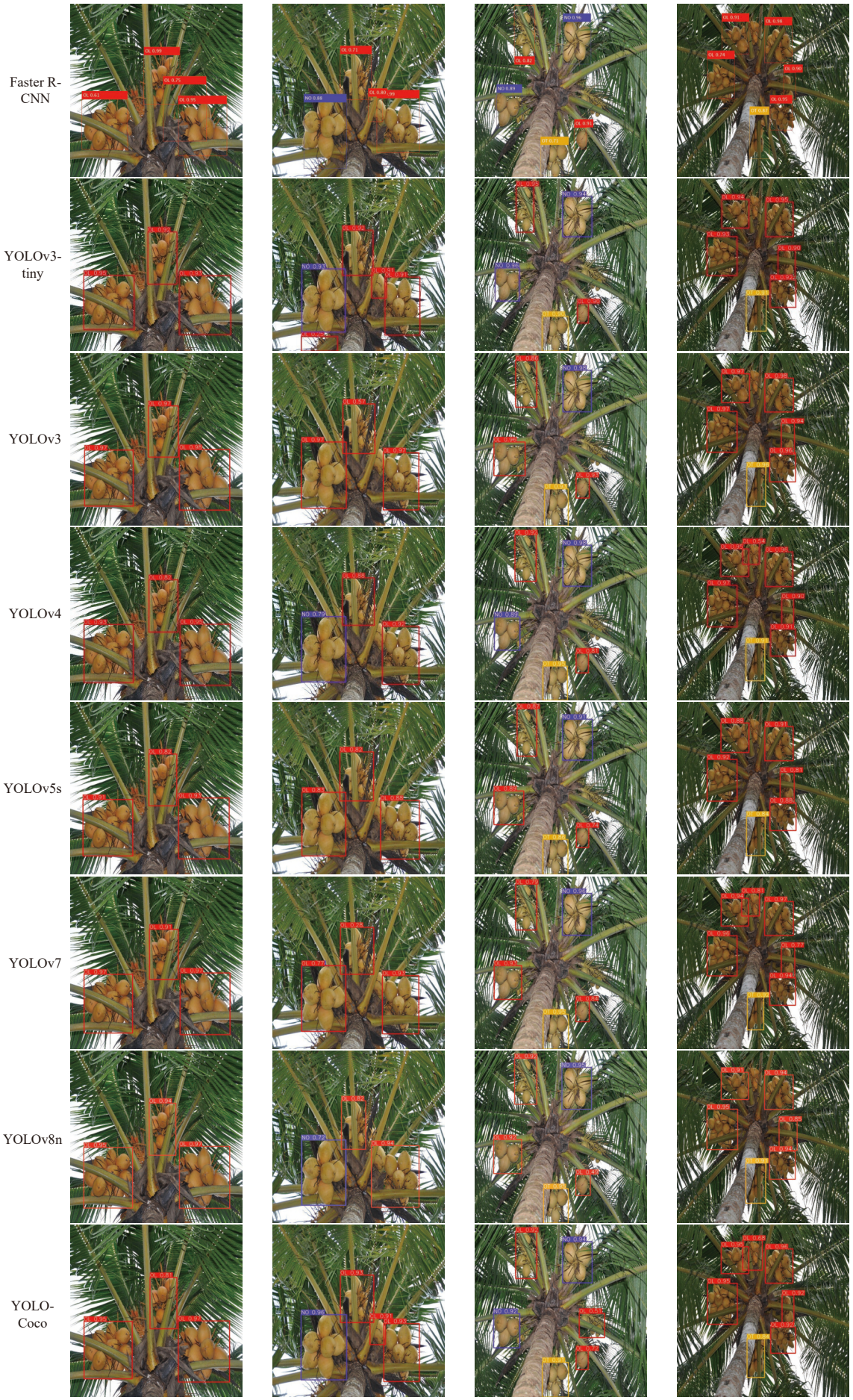


Figure 13 AP results for different classes of coconut clusters detected by different networks

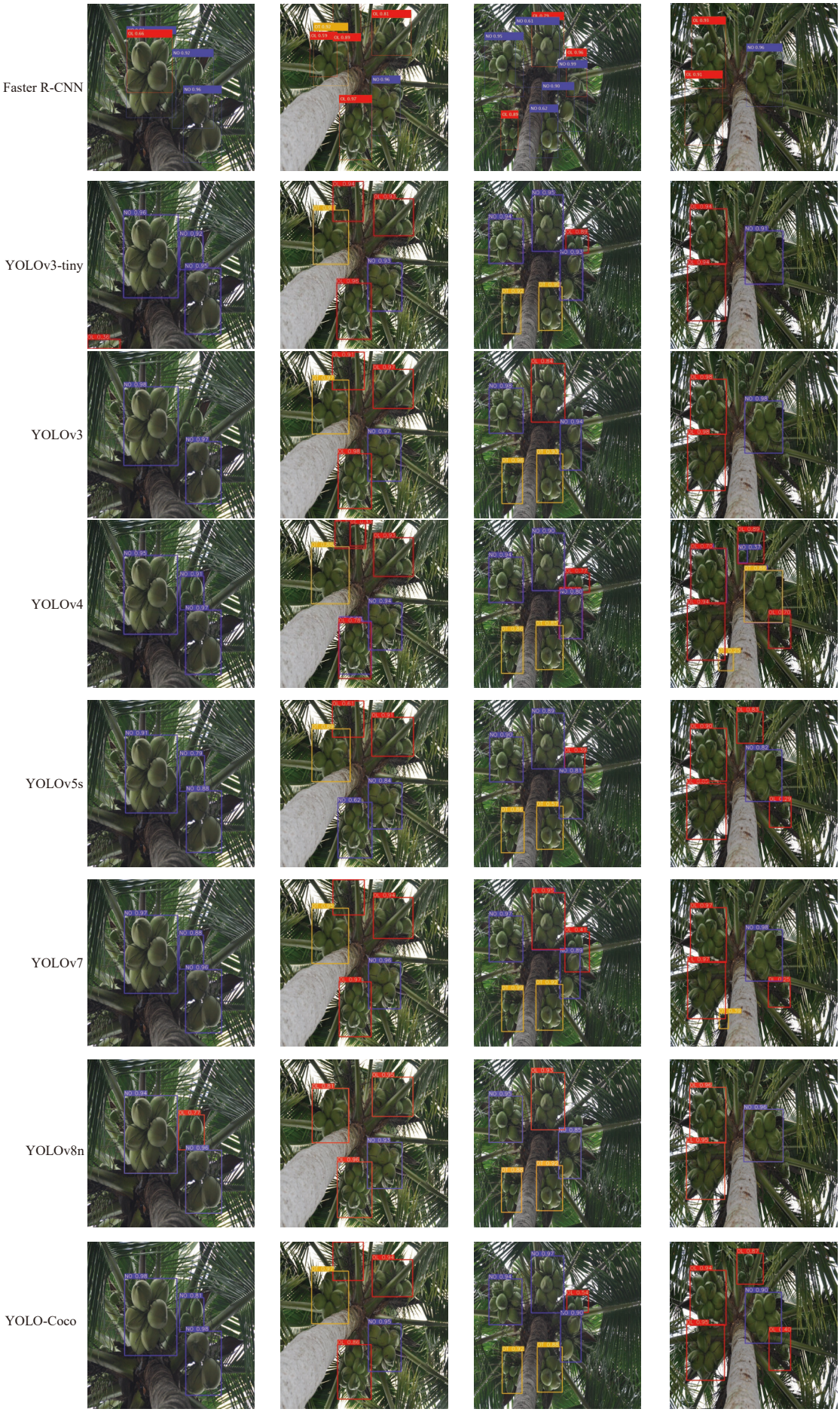
3.5.2 Multi-class detection effect of mainstream detection networks

To more intuitively compare the detection performance of YOLO-Coco and seven mainstream deep learning models, four cases were selected from yellow coconuts and green coconuts, respectively. The detection results of multi-class coconut clusters with different models are shown in Figures 14 and 15. Faster R-CNN shows severe bounding box offset, resulting in incorrect localization of coconut clusters. In addition, there are multiple overlapping bounding boxes in the detection of green coconuts. YOLOv3 and YOLOv3-tiny are able to detect and classify most coconut clusters correctly but pay weak attention to the young coconuts, and there is a leakage of the young fruits. In addition, YOLOv3-tiny had a false detection, detecting the background as a coconut. Due to the similarity in color between green coconuts and leaves, YOLOv4 fails to distinguish the occlusion relationship between leaves and coconuts well in green coconut detection, resulting in overlapping bounding boxes and a cluster of coconuts having two class labels. YOLOv5s produces similar detection results as YOLO-Coco in the image, but the object confidence



Notes: The first and second columns are local canopy scenes. The third and fourth columns are wide-field scenes.

Figure 14 Detection effect of various mainstream detection networks on yellow coconuts



Notes: The first column is the local canopy scene. The other columns are wide-field scenes.

Figure 15 Detection effect of various mainstream detection networks on green coconuts

detected by YOLO-Coco on the same coconut cluster is generally higher than that detected by YOLOv5s. Compared to the state-of-the-art YOLOv7 and YOLOv8n, YOLO-Coco not only detects and correctly classifies all coconut clusters in yellow and green coconuts, but also pays more attention to the features of OL coconut clusters and their young fruits. This is crucial for detecting coconuts in real coconut orchards, as there are more occluded coconuts than non-occluded ones. In conclusion, various performance comparisons indicate that YOLO-Coco has high detection accuracy while maintaining high speed, and is the best detection model for multi-class coconut clusters.

4 Discussion

As known, Parvathi used Faster R-CNN to develop a detection system for the coconut maturity stage. The system detects tender coconuts suitable for drinking and mature coconuts for commercial use. This has inspired us to combine maturity and occlusion conditions to distinguish between tender coconuts suitable for drinking and young fruits in Figures 16a and 16b, thereby providing a better perception of the visual system. However, this may double the number of classes and require more images to train the network model. Divyanth et al.[37] used Faster R-CNN, which added an attention mechanism, to divide coconut clusters into the non-occluded and leaf-occluded coconut targets, with AP being 91.2% and 88.3%, respectively. It takes 0.77s to detect a 512×512 pixel image. AP of the model this study developed was 93.8% on the OL coconut cluster, which is more advantageous for detecting occluded coconuts. It also fits the research purpose because there are more occluded coconuts than non-occluded ones. More importantly, this study defined NO and OL coconut clusters as coconuts in the picking area and OT coconut clusters as coconuts in the waiting picking area. The classification criteria should be more in line with the way the robot works. In addition, both studies used Faster R-CNN, which is slower when detecting coconuts. Fortunately, the model this study developed has faster detection speed and more real-time performance.

a. Different growth periods in yellow coconuts
b. Different growth periods in green coconuts
c. Overlapping bounding boxes in yellow coconuts
d. Overlapping bounding boxes in green coconuts
e. A severely occluded coconut cluster

Notes: Yellow diamond indicates a young coconut cluster; circle indicates an error in coconut detection; yellow triangle indicates a severely occluded coconut cluster.

Figure 16 Examples that need attention in coconut cluster detection

There are a few single coconuts in the dataset, and learning too many single coconut features will cause sparse coconut clusters to be incorrectly detected as multiple labels, as shown in Figures 16c and 16d. A large number of sparse coconut cluster sample features will be provided for model learning so that sparse coconut clusters can be detected more easily. This will further improve the mAP of YOLO-Coco. In conclusion, this study develops a highly accurate multi-class coconut cluster detection model that can help robots to better formulate picking strategies in the canopy. More importantly, the picking location is not easily exposed when the coconut cluster is occluded by multiple leaves, as shown in Figure 16e. Either a special path is adapted so that the end-effector reaches the coconut cluster, or the leaves must be pruned before harvesting to expose its picking point. Therefore, in response to the case where the coconut cluster is occluded by multiple leaves, further studies should also consider using robots to remove the leaves that occlude coconuts, thus simplifying the picking environment.

5 Conclusions

In this study, the fruit detection needs in the robot picking scene were fully considered. Coconuts were divided into three classes, which facilitated the selection strategy that fruit-picking robots may implement. Therefore, this study proposed the YOLO-Coco model for the detection of multi-class coconut clusters. The developed network model used ECA to strengthen the feature weights extracted from the backbone network, and RepConv provided more semantic information for the detection head. Finally, the BiFPN head network carried out the weighted bidirectional fusion of features with different resolutions, which further improved the detection accuracy. In addition, RepConv converted a three-branch structure to a single branch to speed up the inference.

Under the complex canopy environment, the mAP of YOLO-Coco for detection of multi-class coconut clusters was 93.6%, and the AP of not occluded (NO), occluded by leaves (OL), and occluded by trunk (OT) were 90.5%, 93.8%, and 96.4%, respectively. In addition, the detection accuracy of YOLO-Coco in yellow coconuts was 5.1% higher than that in green coconuts. Compared with seven mainstream deep learning networks such as YOLOv8n, the proposed YOLO-Coco achieved the highest mAP value and precision rate with a high detection speed. In addition, YOLO-Coco weight size also showed better advantages, suitable for deployment in embedded devices and mobile terminals.

The high-accuracy detection model of multi-class fruits helps reduce the possibility of end-effector or robot damage. Multi-class detection results can be further used to develop the fruit picking order and path, such as the preference for picking NO over OL in the picking area. In the future, the depth information will be combined to propose a picking strategy suitable for coconut harvesting and provide the robot with a full-view picking path of the canopy. This will enable the picking robot to achieve fast and accurate picking operations, providing strong support for automated coconut picking.

Acknowledgments

This work was financially supported by the Key R&D Projects in Hainan Province (Grant No. ZDYF2022XDNY231), the National Natural Science Foundation of China (Grant No. 52265040), and the Innovative Research Projects for Graduate Students in Hainan Province (Grant No. Qhyb2023-100).

[References]

[1] Mat K, Abdul Kari Z, Rusli N D, Che Harun H, Wei L S, Rahman M M, et al. Coconut palm: Food, feed, and nutraceutical properties. Animals, 2022; 12(16): 2107.

[2] Ru S F, Wang J F, Fan J Q. Design and parameter optimization of removing coconut fiber device by bionic steel wire roller brush based on characteristics of claw-toe. Transactions of the CSAE, 2018; 22(34): 27–35. (in Chinese)

[3] Hainan Provincial Bureau of Statistics. Hainan Statistical Yearbook, 2023. Available: https://stats.hainan.gov.cn/tjj/tjsu/ndsj/2023/202311/P0202311-29587527396831.pdf. Accessed on [2023-12-02].

[4] Caladcad J A, Cabahug S, Catamco M R, Villaceran P E, Cosgafa L, Cabizares K N, et al. Determining Philippine coconut maturity level using machine learning algorithms based on acoustic signal. Computers and Electronics in Agriculture, 2020; 172: 105327.

[5] Ignacio I F, Miguel T S. Research opportunities on the coconut (*Cocos nucifera* L.) using new technologies. South African Journal of Botany, 2021; 141: 414–420.

[6] Zhou H Y, Wang X, Au W, Kang H W, Chen C. Intelligent robots for fruit harvesting: Recent developments and future challenges. Precision Agriculture, 2022; 23(5): 1856–1907.

[7] Gené-Mola J, Vilaplana V, Rosell-Polo J R, Morros J R, Ruiz-Hidalgo J, Gregorio E. Multi-modal deep learning for Fuji apple detection using RGB-D cameras and their radiometric capabilities. Computers and Electronics in Agriculture, 2019; 162: 689–698.

[8] Megalingam R K, Manoharan S K, Mohandas S M, Vadivel S R R, Gangireddy R, Ghanta S, et al. Amaran: An unmanned robotic coconut tree climber and harvester. IEEE-ASME Transactions on Mechatronics, 2021; 26(1): 288–299.

[9] Wibowo T S, Sulistijono I A, Risnumawan A. End-to-end coconut harvesting robot. In: 2016 International Electronics Symposium (IES), 2016; pp.444–449. doi: 10.1109/ELECSYM.2016.7861047.

[10] Parvathi S, Selvi S T. Detection of maturity stages of coconuts in complex background using Faster R-CNN model. Biosystems Engineering, 2021; 202: 119–132.

[11] Fu L S, Majeed Y, Zhang X, Karkee M, Zhang Q. Faster R-CNN-based apple detection in dense-foliage fruiting-wall trees using RGB and depth features for robotic harvesting. Biosystems Engineering, 2020; 197: 245–256.

[12] Tang Y C, Zhou H, Wang H J, Zhang Y Q. Fruit detection and positioning technology for a *Camellia oleifera* C. Abel orchard based on improved YOLOv4-tiny model and binocular stereo vision. Expert Systems with Applications, 2023; 211: 118573.

[13] Zhu X Y, Chen F J, Zhang X W, Zheng Y L, Peng X D, Chen C. Detection the maturity of multi-cultivar olive fruit in orchard environments based on Olive-EfficientDet. Scientia Horticulturae, 2024; 324: 112607.

[14] Patrício D I, Rieder R. Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review. Computers and Electronics in Agriculture, 2018; 153: 69–81.

[15] Shaikh T A, Rasool T, Lone F R. Towards leveraging the role of machine learning and artificial intelligence in precision agriculture and smart farming. Computers and Electronics in Agriculture, 2022; 198: 107119.

[16] Zhu X Y, Li J J, Jia R C, Liu B, Yao Z H, Yuan A H, et al. LAD-Net: A novel light weight model for early apple leaf pests and diseases classification. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2023; 20(2): 1156–1169.

[17] Wang Z, Hua Z X, Wen Y C, Zhang S J, Xu X S, Song H B. E-YOLO: Recognition of estrus cow based on improved YOLOv8n model. Expert Systems with Applications, 2024; 238: 122212.

[18] Montoya-Cavero L E, Torres R D D, Gómez-Espinosa A, Cabello J A E. Vision systems for harvesting robots: Produce detection and localization. Computers and Electronics in Agriculture, 2022; 192: 106562.

[19] Yang Y Y, Han Y X, Li S, Yang Y D, Zhang M, Li H. Vision based fruit recognition and positioning technology for harvesting robots. Computers and Electronics in Agriculture, 2023; 213: 108258.

[20] Zhang B, Wang R R, Zhang H M, Yin C H, Xia Y Y, Fu M, et al. Dragon fruit detection in natural orchard environment by integrating lightweight network and attention mechanism. Frontiers in Plant Science, 2022; 13: 1040923.

[21] Zhang J, Karkee M, Zhang Q, Zhang X, Yaqoob M, Fu L S, et al. Multi-class object detection using faster R-CNN and estimation of shaking locations for automated shake-and-catch apple harvesting. Computers and Electronics in Agriculture, 2020; 173: 105384.

[22] Gao F, Fu L S, Zhang X, Majeed Y, Li R, Karkee M, Zhang Q. Multi-class fruit-on-plant detection for apple in SNAP system using Faster R-CNN. Computers and Electronics in Agriculture, 2020; 176: 105634.

[23] Song H B, Shang Y Y, He D J. Review on deep learning technology for fruit target recognition. Transactions of the CSAM, 2023; 54(1): 1–19. (in Chinese)

[24] Peng H X, Huang B, Shao Y Y, Li Z S, Zhang C W, Chen Y, et al. General improved SSD model for picking object recognition of multiple fruits in natural environment. Transactions of the CSAE, 2018; 16(34): 155–162. (in Chinese)

[25] Redmon J, Divvala S, Girshick R, Farhadi A. You Only Look Once: Unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas: IEEE, 2016; pp.779–788. doi: 10.1109/CVPR.2016.91.

[26] Li T H, Sun M, He Q H, Zhang G S, Shi G, Ding X, et al. Tomato recognition and location algorithm based on improved YOLOv5. Computers and Electronics in Agriculture, 2023; 208: 107759.

[27] Xu L J, Wang Y H, Shi X S, Tang Z L, Chen X Y, Wang Y C, et al. Real-time and accurate detection of citrus in complex scenes based on HPL-YOLOv4. Computers and Electronics in Agriculture, 2023; 205: 107590.

[28] Suo R, Gao F F, Zhou Z X, Fu L S, Song Z Z, Dhupia J, et al. Improved multi-classes kiwifruit detection in orchard to avoid collisions during robotic picking. Computers and Electronics in Agriculture, 2021; 182: 106052.

[29] Zhang F, Chen Z J, Ali S, Yang N, Fu S L, Zhang Y K. Multi-class detection of cherry tomatoes using improved YOLOv4-Tiny. Int J Agric & Biol Eng, 2023; 16(2): 225–231.

[30] Wang C Y, Bochkovskiy A, Liao H Y M. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. In: 2023 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2023; pp.7464–7475. doi: 10.1109/CVPR52729.2023.00721.

[31] Wang Y, Yao X Z, Li B, Xu S, Yi Z F, Zhao J H. Malformed sweet pepper fruit identification algorithm based on improved YOLOv7-tiny. Transactions of the CSAM, 2023; 11(54): 236–246. (in Chinese)

[32] Wang Q L, Wu B G, Zhu P F, Li P H, Zuo W M, Hu Q H. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In: 2020 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020; pp.11531–11539. doi: 10.1109/CVPR42600.2020.01155.

[33] Li Q M, Han Z C, Wu X M. Deeper insights into graph convolutional networks for semi-supervised learning. In: 32nd AAAI Conference on Artificial Intelligence, 2018; pp.3538–3545. doi: 10.48550/arXiv.1801.07606.

[34] Ding X H, Zhang X Y, Ma N N, Han J G, Ding G G, Sun J. RepVGG: Making VGG-style ConvNets Great Again. In: 2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021; pp.13728–13737. doi: 10.1109/CVPR46437.2021.01352.

[35] Tan M X, Pang R Y, Le Q V. EfficientDet: Scalable and Efficient Object Detection. In: 2020 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020; pp.10778–10787. doi: 10.1109/CVPR42600.2020.01079.

[36] Yao T, Tan Z P, Cheng E, Wu L G. Method for daylily detection and segmentation based on improved YOLOv7-seg. Transactions of the CSAE, 2024; 40(9): 146–153. (in Chinese)

[37] Divyanth L G, Soni P, Pareek C M, Machavaram R, Nadimi M, Paliwal J, et al. Detection of coconut clusters based on occlusion condition using attention-guided Faster R-CNN for robotic harvesting. Foods, 2022; 11(23): 3903.