Article

Apple Detection in Complex Scene Using the Improved YOLOv4 Model

Lin Wu, Jie Ma *, Yuehua Zhao and Hong Liu

School of Electronics and Information Engineering, Hebei University of Technology, Tianjin 300401, China; wulin248@gmail.com (L.W.); cassiel0406@gmail.com (Y.Z.); 201821902006@stu.hebut.edu.cn (H.L.)

* Correspondence: jma@hebut.edu.cn; Tel.: +86-139-2038-1537

Abstract: To enable the apple picking robot to quickly and accurately detect apples under the complex background in orchards, we propose an improved You Only Look Once version 4 (YOLOv4) model and data augmentation methods. Firstly, the crawler technology is utilized to collect pertinent apple images from the Internet for labeling. For the problem of insufficient image data caused by the random occlusion between leaves, in addition to traditional data augmentation techniques, a leaf illustration data augmentation method is proposed in this paper to accomplish data augmentation. Secondly, due to the large size and calculation of the YOLOv4 model, the backbone network Cross Stage Partial Darknet53 (CSPDarknet53) of the YOLOv4 model is replaced by EfficientNet, and convolution layer (Conv2D) is added to the three outputs to further adjust and extract the features, which make the model lighter and reduce the computational complexity. Finally, the apple detection experiment is performed on 2670 expanded samples. The test results show that the EfficientNet-B0-YOLOv4 model proposed in this paper has better detection performance than YOLOv3, YOLOv4, and Faster R-CNN with ResNet, which are state-of-the-art apple detection model. The average values of Recall, Precision, and F1 reach 97.43%, 95.52%, and 96.54% respectively, the average detection time per frame of the model is 0.338 s, which proves that the proposed method can be well applied in the vision system of picking robots in the apple industry.

Keywords: apple detection; YOLOv4; EfficientNet; picking robot; data augmentation

Citation: Wu, L.; Ma, J.; Zhao, Y.; Liu, H. Apple Detection in Complex Scene Using the Improved YOLOv4 Model. *Agronomy* **2021**, *11*, 476. <https://doi.org/10.3390/agronomy11030476>

Academic Editors: Karsten Schmidt and Silvia Arazuri

Received: 10 January 2021

Accepted: 2 March 2021

Published: 4 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction and Related Works

Apple is one of the most popular fruits, and its output is also among the top three in global fruit sales. According to incomplete statistics, there are more than 7500 types of known apples [1] in the world. However, experienced farmers are still the main force of agricultural production. Manual work consumes time and increases production costs, and workers who lack knowledge and experience will make unnecessary mistakes. With the continuous progress of precision agricultural technology, fruit picking robots have been widely used in agriculture. In the picking systems, there are mainly two subsystems: the vision system and the manipulator system [2]. The vision system detects and localizes fruits and guides the manipulator to detach fruits from trees. Therefore, a robust and efficient vision system is the key to the success of the picking robot, but due to the complex background in orchards, there are still many challenges in this research.

For the complex background in orchards, the dense occlusion between leaves is one of the biggest interference factors in apple detection, which will cause false detection or missed detection of apples. Therefore, to make the model learn features better, the training data should contain more comprehensive scenes. However, due to the huge number of apples and complex background, apple labeling is a very time-consuming and energy-consuming task, which leads to the number of most datasets ranges from dozens to thousands of images [3–7], and covers a single scene. The data for occlusion scenes is even scarcer, which is not conducive to enhancing the detection ability of the model. To overcome this deficiency, we propose a leaf illustration data augmentation method to expand the

dataset. To further expand the number of the dataset and enrich the complexity of the scene, common data augmentation methods such as mirror, crop, brightness, blur, dropout, rotation, scale, and translation are also utilized in this paper. The experimental results show that the model trained by traditional augmentation techniques and an illustration augmentation technique proposed in this paper can well detect apples under complex scenes in orchards.

In recent years, the research on apple detection under complex scenes in orchards has also made some progress. Tian Y et al. [8] proposed an improved YOLOv3 model to detect apples in different growth periods in orchards, with the F1 score of 0.817. Kang H et al. [9] proposed a new LedNet model and an automatic labeling tool, with the Recall and the accuracy at 0.821 and 0.853, respectively. Mazzia V et al. [10] used the YOLOv3-tiny model to match the embedded device, which achieved the detection speed of 30 fps without affecting the mean Average Precision (mAP) (83.64%). Kuznetsova A et al. [11] proposed pre-processing and post-processing operations to adapt to the YOLOv3 model, the detection result shown that the average detection time was 19 ms, 7.8% of the objects were mistaken and 9.2% of apples were not recognized for apples. Gao F et al. [12] used Faster Regions with Convolutional Neural Networks (Faster R-CNN) to detect apples in dense-foliage fruiting-wall trees, the experimental result was that the mAP was 0.879 and the average detection time was 0.241 s, which effectively detected apples under various occlusion conditions; Liu X et al. [13] proposed an apple detection based on color and shape features method, the detection results were that the value of Recall, Precision, and F1 score reached 89.80%, 95.12%, and 92.38%, respectively. Jia W et al. [14] combined ResNet and DenseNet to improve Mask R-CNN, which reduced the input parameters, with the Precision of 97.31% and the Recall of 95.70%.

For picking robots, the model should have fast and accurate detection performance. The YOLO [15–18] models unify object classification and object detection into a regression problem. The YOLO models do not use the area proposal process but directly use regression to detect objects. Therefore, the detection process is effectively accelerated. Compared with the YOLOv3 model, the latest version YOLOv4 model owns better accuracy under maintaining the same speed. However, the YOLOv4 model has not been widely used for fruit detection. Due to the large size and computational complexity of the YOLOv4 model, it is a huge burden for low-performance devices. EfficientNet [19] uses a compound coefficient to balance the three dimensions (depth, width, and resolution) of the model on limited resources, which can maximize the accuracy of the model. Therefore, we utilize EfficientNet to replace the backbone network CSPDarknet53 of the YOLOv4 model, and Conv2D is added to the three outputs to further extract and adjust the features, which can make the improved model lighter and better detection performance. The experimental results show that the improved model can be well applied to the vision system of the picking robot.

The rest of this paper is organized as follows. Section 2 introduces the dataset collection, common data augmentation methods, and the proposed illustration data augmentation method. Section 3 introduces YOLOv4, EfficientNet, and the improved EfficientNet-B0-YOLOv4 model. Section 4 is experimental configuration, experimental results, and discussion. Finally, the conclusions and prospects of this paper are described.

2. Dataset and Data Augmentation

2.1. Dataset

In this paper, we choose Red Fuji apple as the experimental object. Since there are a large number of apple-related images on the Internet, we use the Python language to develop an image crawler to download these images in batches, which reduces the cost of data collection and improves the efficiency of data collection.

The main sources of images are Baidu and Google. The search keywords are Red Fuji Apple, Apple Tree, and Apple, etc. Firstly, to ensure image quality, the width or height of the crawled image is set to be at least greater than 500 pixels. Secondly, after

the manual screening, the repetitive, fuzzy, and inconsistent images are mainly removed. Finally, 267 high-quality images are obtained, of which 35 images contain only a single apple, 54 images with multiple apples without overlapping, and 178 images with multiple apples overlapping.

Then, these 267 images are expanded to 2670 images by using data augmentation methods. Randomly divide 1922 images as the training set to train the detection model, 214 images as the validation set to adjust the model parameters, and 534 images as the test set to verify the detection performance. To better compare the performance of different models, images in the training set are converted to PASCAL VOC format. The completed dataset is shown in Table 1.

Table 1. The number of apple images generated by data augmentation methods.

Original	Mirror	Crop	Brightness	Blur	Dropout	Rotation	Scale	Translation	Illustration	Total
267	267	267	267	267	267	267	267	267	267	2670

2.2. Common Data Augmentation

In this paper, we use 8 common data augmentation methods to expand the dataset, respectively mirror, crop, brightness, blur, dropout, rotation, scale, and translation operation. These operations are used to further simulate the complex scenes of apple detection in orchards. Figure 1b–i shows the effects of various common data augmentation after these operations.

Figure 1. Common data augmentation methods: (a) original image; (b) horizontal mirror; (c) crop processing; (d) brightness transformation; (e) gaussian blur processing; (f) dropout processing; (g) rotation processing; (h) scale processing; (i) translation processing.

2.2.1. Image Mirror

In orchards, the position and direction of the apples are various. Therefore, we use 50% probability horizontal mirroring and 50% probability vertical mirroring to process the original image. Both can be used alone or in combination.

2.2.2. Image Crop

In many apples stacked together, there will be various occlusion problems, and some apples will be blocked a little or more. Therefore, we randomly cut off 20% of the original image edges to simulate this scene.

2.2.3. Image Brightness

When the illumination is strong or weak, it will lead to apple color changes, which cause huge interference for the detection. Therefore, to enhance the robustness of the model, we randomly multiply the image with a brightness factor between 0.5 and 1.5.

2.2.4. Image Blur

Sometimes the image captured by the picking robot may be unclear or blurred, which can also cause interference with the apple detection. Therefore, we use the gaussian blur with a mean value of 2.0 and a standard deviation of 8.0 to augment the dataset.

2.2.5. Image Dropout

Apples often encounter the problem of diseases and insect pests, typically leaving traces of numerous spots. Therefore, we randomly dropout the grid points between 0.01 and 0.1 on the original image, and the grid points are filled with black.

2.2.6. Image Rotation

Similar to the mirror method, rotation is to further increase the image viewing angles. Therefore, we use randomly rotating the original image at an angle between -30° and 30° to augment the dataset, and the space vacated by the rotation is filled with black.

2.2.7. Image Scale

Due to the different positions of the apples in orchards, there will be apples of different sizes when capturing images. Therefore, to simulate this scene, we randomly multiply the original image with a scaling factor between 0.5 and 1.5.

2.2.8. Image Translation

Similar to the crop method, translation is to further solve the occlusion problem of the apple. Therefore, we randomly translate 20% of the edges of the original image, and the space after translation is filled with black.

2.3. Illustration Data Augmentation

To enrich the background and texture of training images, in Mixup [20], CutMix [21], Mosaic [18], several original images can be regularly mixed or superimposed to form a new image. For example, four original images are evenly spread into four-square-grid images, which will not increase the cost of training and inference but enhance the localization ability of the model. For apple detection in orchards, the biggest interference to the apple detection is the dense occlusion between leaves, and the complexity and randomness of the background, which makes feature extraction difficult and leads to false detection or missed detection of apples.

Therefore, to simulate the complexity of the scene and enrich the background of the object, we propose a leaf illustration data augmentation method, which uses some leaf illustrations to randomly insert on the original image. Firstly, collect 5 kinds of apple leaf illustrations, as shown in Figure 2. The format of the illustration is PNG, only contains the object itself, and the background is transparent, which helps protect the original image after insertion and avoid adding the invalid background. Secondly, the illustration size is 1/8 to 1/4 of the average value of all the ground-truth in the current image, and the number of insertions is 5 to 15 times. Finally, the original dataset is expanded in batches by using the illustration data augmentation method. Figure 3 shows the augmentation effects of different leaf illustrations.

Figure 2. Five kinds of apple leaves.

Figure 3. (a) Original image; (b) leaf data augmentation.

3. Methodologies

3.1. YOLOv4

YOLOv4 is the state-of-the-art, real-time detection model, which is further improved based on the YOLOv3 model. As a result, on the MS COCO dataset, without a drop in the frames per second (FPS), the mean Average Precision (mAP) is increased to 44%, and the overall performance is significantly improved. There are three major improvements in the network structure: (1) Using the CSPNet [22] to modify Darknet53 to CSPDarknet53, which further promotes the fusion of low-level information and achieves stronger feature extraction capabilities. As shown in Figure 4b, the original residual module is divided into left and right parts. The right parts maintain the original residual stack, and the left parts use a large residual edge to fuse the low-level information with the high-level information extracted from the residual block; (2) Using the Spatial Pyramid Pooling (SPP) [23] to add 4 different max-pooling operations at the last output to further extract and fuse features, the convolution kernel size is (1×1) , (5×5) , (9×9) , and (13×13) , as shown in Figure 5; (3) Modifying Feature Pyramid Networks (FPN) [24] structure to Path Aggregation Network (PANet) [25], that is, add a top-down structure to the bottom-up structure of FPN to further extract and merge feature information, as shown in Figure 6b.

Figure 4. Comparison between Darknet53 and CSPDarknet53. (a) Darknet53, (b) CSPDarknet53.

Figure 5. SPP.

Figure 6. Comparison between FPN and PANet. (a) FPN, (b) PANet.

3.2. EfficientNet

In recent years, the rapid development of deep learning has spawned various excellent convolutional neural networks. From the initial simple network [26–28] to the current complex network [29–32], the performance of the models is getting better and better in all aspects. EfficientNet combines the advantages of previous networks, which summarizes the improvement of network performance into three dimensions: (1) Deepen the network, that is, use the skip connection to increase the depth of the neural network, and achieve feature extraction through deeper layers; (2) Widen the network, that is, increase the number of convolutional layers to achieve more features and obtain more functions; (3) By increasing the input image resolution, the network can learn and express more things, which is beneficial to improve accuracy. Then, use a compound coefficient ϕ to uniformly scale and balance the depth, width, and resolution of the network, and maximize the network accuracy on limited resources. The calculation of the compound coefficient is shown in Equation (1):

$$\begin{aligned}
 \text{depth} : d &= \alpha^\phi \\
 \text{width} : w &= \beta^\phi \\
 \text{resolution} : r &= \gamma^\phi \\
 \text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 &\approx 2 \\
 \alpha \geq 1, \beta &\geq 1, \gamma \geq 1
 \end{aligned} \tag{1}$$

where the d , w , and r are the coefficients used to scale the depth, width, and resolution of the network. The α , β , and γ are resource allocation for network depth, width, and resolution. According to the research of Tan M [19] in his paper, the network parameters of EfficientNet-B0 are shown in Table 2. The optimal coefficients of the network are: $\alpha = 1.2$, $\beta = 1.1$, $\gamma = 1.15$. EfficientNet is mainly made up of Stem, 16 Blocks, Conv2D, GlobalAveragePooling2D, and Dense layers. The design of Blocks is mainly based on the residual structure and attention mechanism, and the other structures are similar to

conventional convolutional neural networks. Figure 7 shows the EfficientNet-B0, which is the baseline network of EfficientNet.

Table 2. EfficientNet-B0 network parameter.

Stage i	Operator $\hat{F}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3 × 3	224 × 224	32	1
2	MBConv1, k3 × 3	112 × 112	16	1
3	MBConv6, k3 × 3	112 × 112	24	2
4	MBConv6, k5 × 5	56 × 56	40	2
5	MBConv6, k3 × 3	28 × 28	80	3
6	MBConv6, k5 × 5	14 × 14	112	3
7	MBConv6, k5 × 5	14 × 14	192	4
8	MBConv6, k3 × 3	7 × 7	320	1
9	Conv1 × 1 & Pooling & FC	7 × 7	1280	1

Figure 7. EfficientNet-B0.

3.3. EfficientNet-B0-YOLOv4

There are 8 versions of EfficientNet (B0–B7). With the increase of the version, the performance of the model gradually improves, but the corresponding model size and calculation amount also gradually increases. Although the original YOLOv4 model has excellent performance, its size and calculation amount are large, which is not suitable for the application of some low-performance devices. To further improve the accuracy and efficiency of the YOLOv4 model and consider the size of the model, we replace the backbone network CSPDarknet53 of the YOLOv4 model with EfficientNet-B0, and choose P3, P5, and P7 as three different feature layers. Since the output sizes of the three feature layers of the CSPDarknet53 are (256 × 256), (512 × 512), and (1024 × 1024), respectively, the corresponding P3 is (40 × 40), P5 is (112 × 112), and P7 is (320 × 320). Therefore, to match the size and further extract the features, Conv2D is added to adjust the three output features. Figure 8 shows the network structure of EfficientNet-YOLOv4.

Figure 8. EfficientNet-YOLOv4.

The loss function remains the same as the YOLOv4 model, which consists of three parts: classification loss, regression loss, and confidence loss. Classification loss and confidence loss remain the same as the YOLOv3 model, but Complete Intersection over Union (CIoU) [33] is used to replace mean squared error (MSE) to optimize the regression loss.

The CIoU loss function is as follows:

$$LOSS_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v, \quad (2)$$

where $\rho^2(b, b^{gt})$ represents the Euclidean distance between the center points of the prediction box and the ground truth, c represents the diagonal distance of the smallest closed area that can simultaneously contain the prediction box and the ground truth. Figure 9 shows the structure of CIoU.

Figure 9. CIoU.

The formulas of α and v are as follows:

$$\alpha = \frac{v}{1 - IoU + v}, \quad (3)$$

$$v = \frac{4}{\pi^2} \left(\arctan \frac{w^{gt}}{h^{gt}} - \arctan \frac{w}{h} \right)^2. \quad (4)$$

The total loss function of the YOLOv4 model is:

$$\begin{aligned}
 LOSS = & 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v - \\
 & \sum_{i=0}^{S^2} \sum_{j=0}^B I_{ij}^{obj} \left[\hat{C}_i \log(C_i) + (1 - \hat{C}_i) \log(1 - C_i) \right] - \\
 & \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^B I_{ij}^{noobj} \left[\hat{C}_i \log(C_i) + (1 - \hat{C}_i) \log(1 - C_i) \right] - \\
 & \sum_{i=0}^{S^2} I_{ij}^{obj} \sum_{c \in classes} \left[\hat{p}_i(c) \log(p_i(c)) + (1 - \hat{p}_i(c)) \log(1 - p_i(c)) \right]
 \end{aligned} \quad (5)$$

where S^2 represents $S \times S$ grids, each grid generates B candidate boxes, and each candidate box gets corresponding bounding boxes through the network, finally, $S \times S \times B$ bounding boxes are formed. If there is no object (noobj) in the box, only the confidence loss of the box is calculated. The confidence loss function uses cross entropy error and is divided into two parts: there is the object (obj) and noobj. The loss of noobj increases the weight coefficient λ , which is to reduce the contribution weight of the noobj calculation part. The classification loss function also uses cross entropy error. When the j -th anchor box of the i -th grid is responsible for certain ground truth, then the bounding box generated by this anchor box will calculate the classification loss function.

4. Experiments and Discussion

4.1. Experimental Details

4.1.1. Simulation Setup

The experimental environment of this paper is under Ubuntu 18.04 system, GPU is Tesla K80 (12 GB), CPU is Intel XEON, and the models are all written with PyTorch. General settings: the training epoch is 100, the learning rate for the first 50 epochs is 1×10^{-3} and the batch size is 16, the learning rate for the next 50 epochs is 1×10^{-3} and the batch size is 8. Due to the relatively small RAM, the input image size of the YOLOv4 model is changed from 608×608 to 416×416 , which is the same as the original YOLOv3 model. Table 3 shows the basic configuration of the local computer.

Table 3. The basic configuration of the local computer.

Computer Configuration	Specific Parameters
CPU	Intel XEON
GPU	Tesla K80
Operating system	Ubuntu18.04
Random Access Memory	12 GB

4.1.2. Evaluation Index

In the binary classification problem, according to the combination of the sample's true class and the model's prediction class, it can be divided into 4 types: TP, FP, TN, and FN. TP means true positive, that is, the actual is positive and the prediction is also positive; FP means false positive, that is, the actual is negative but the prediction is positive; TN is true negative, that is, the actual is negative and the prediction is also negative; FN means false negative, that is, the actual is positive but the prediction is negative. The Precision represents the proportion of samples that are true positive among all samples predicted to be positive. The Recall represents the proportion of samples predicted to be positive among the samples that are true positive. The AP value of each class is the area under the P-R curve formed by the Precision and the Recall. The mAP value is the average of the AP values of all classes. The F1 score is based on the harmonic average of the Precision and the Recall. The formula definition is as follows:

Precision:

$$P = \frac{TP}{TP + FP}. \quad (6)$$

Recall:

$$R = \frac{TP}{TP + FN}. \quad (7)$$

Average Precision:

$$AP = \int_0^1 p(r)dr. \quad (8)$$

Mean Average Precision:

$$mAP = \frac{1}{n} \sum_{i=1}^n AP_i. \quad (9)$$

F1 score:

$$F1 = 2 \times \frac{P \times R}{P + R}. \quad (10)$$

4.2. Experimental Results

4.2.1. Influence of Data Augmentation Methods

To obtain better detection results, traditional data augmentation techniques and a leaf illustration augmentation technique are adopted to expand the dataset. To evaluate the influence of the augmentation techniques on the EfficientNet-B0-YOLOv4 model, the control variate technique is adopted to get rid of one data augmentation approach every time and get the F1 indicators in the absence of this method, as shown in Table 4. It can be seen from the experimental results that the removal of the illustration data augmentation method has the greatest impact on the performance of the model, and the F1 drops by 2.05%, indicating that the image data augmented by the illustration data augmentation method has a greater contribution to enriching the diversity of the training set. Compared with common data augmentation methods, illustration data augmentation will generate new background and texture for the image, which is of great help to enhance the robustness of model detection, especially under the interference of dense leaves. Due to the complexity and diversity of leaves, the proposed illustration data augmentation method combined with common data augmentation methods to generate apple images can make up for the lack of training images, greatly reduce the workload of labeling, and achieve better results in model detection.

Table 4. F1 comparison between different data augmentation methods.

Data Augmentation Methods										
Illustration	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Translation	✓	✓	✓	✓	✓	✓	✓	✓		
Scale	✓	✓	✓	✓	✓	✓	✓			
Rotation	✓	✓	✓	✓	✓	✓				
Dropout	✓	✓	✓	✓	✓					
Blur	✓	✓	✓	✓						
Brightness	✓	✓	✓							
Crop	✓	✓								
Mirror	✓									
F1	96.54%	95.62%	95.00%	94.87%	94.43%	94.28%	94.01%	93.52%	92.81%	90.76%

To better assess the performance of the improved model, we count the detection results of original images and augmented images. The test results are shown in Table 5. It can be seen that the EfficientNet-B0-YOLOv4 model in this paper can achieve desirable detection results for the apple images augmented by using data augmentation methods. Compared with the original image, the methods of mirror, crop, rotation, scale, and translation are mainly based on the change of image position or angle, which hardly adds new texture

information, so the detection results are similar to those of the original images. The methods of brightness, blur, dropout, and illustration bring new texture information to the image. Although it will cause more false detections, keeps the number of missed detections similar to the original images, and has more object detections, which shows that the rich background will enhance the learning ability of the model. Compared with the detection results of traditional augmentation techniques, the proposed illustration augmentation technique will lead to more false detections, but the detection quantity and missed detections are consistent with these methods, which shows the effectiveness and feasibility of our proposed method.

Table 5. Detection results of original and generated images.

Apple Images	Original	Mirror	Crop	Brightness	Blur	Dropout	Rotation	Scale	Translation	Illustration
Number of detected objects	263	270	266	281	276	274	268	266	265	283
Number of missed objects	3	8	8	6	8	8	9	7	4	7
Number of false objects	32	44	40	53	50	48	43	39	35	56

To further verify the influence of the proposed illustration data augmentation method on the improved model, as shown in Figure 10, the detection results of the model trained by the illustration augmented image are compared with the model trained by the real image. From the detection results, it can be seen that the model trained by the illustration augmented images and the model trained by the real images both can accurately detect apples under no occlusion between leaves. However, under the dense occlusion between leaves, the model trained by real images hardly detects apples, while the model trained by illustration augmented images can detect apples well and improve the detection results. It can be seen that the illustration augmentation method enriches the leaf occlusion scene in the training images, provides richer features for the learning of the model, and thus helps to improve the learning ability and detection results of the model.

Figure 10. Detection results between illustration augmented image and real image: (a) real image; (b) no illustration augmented image; (c) illustration augmented image.

4.2.2. Comparison of Different Models

To verify the superiority of the proposed EfficientNet-B0-YOLOv4 model in this paper, we compare it with YOLOv3, YOLOv4, and Faster R-CNN with ResNet, which are the state-of-the-art apple detection models. Table 6 shows the comparison of F1, mAP, Precision and Recall of different models. Table 7 shows the comparison of the average detection time per frame, weight size, parameter amount and calculation amount (FLOPs) of different models. Table 8 shows the detection results of different models in the test set. Figure 11 shows the comparison of the P-R curve of different models. Figure 12 shows the detection results of different models, where the green ring represents the missed detection and the blue ring represents the false detection.

Table 6. Performance comparison between different models.

Different Models	F1	mAP	Precision	Recall
YOLOv3	91.76%	93.98%	95.81%	88.03%
YOLOv4	96.36%	96.85%	98.22%	94.57%
Faster R-CNN with ResNet	66.96%	82.69%	52.07%	93.76%
EfficientNet-B0-YOLOv4	96.54%	98.15%	95.52%	97.43%

Table 7. Weight comparison between different models.

Different Models	Time/s	Size/MB	Parameter	FLOPs
YOLOv3	0.405	235	6.15×10^7	3.28×10^{10}
YOLOv4	0.410	244	6.40×10^7	2.99×10^{10}
Faster R-CNN with ResNet	6.167	108	2.83×10^7	1.79×10^{11}
EfficientNet-B0-YOLOv4	0.338	158	3.78×10^7	1.27×10^{10}

Table 8. Detection results of different models.

Different Models	Ground-Truth	Faster R-CNN with ResNet	YOLOv3	YOLOv4	EfficientNet-B0-YOLOv4
Number of detected objects	2340	4214	3745	3004	2712
Number of missed objects	0	147	112	45	68
Number of wrong objects	0	2021	1517	709	440

Figure 11. P-R curves of different detection models.

Figure 12. Detection results of different models: (a) YOLOv3; (b) YOLOv4; (c) Faster R-CNN with ResNet; (d) EfficientNet-B0-YOLOv4.

Generally, to make robots pick more real apples in orchards, more attention should be paid to the improvement of the Recall. It can be seen from Tables 6 and 7 that the Faster R-CNN with ResNet model has a better Recall (93.76%), but the other performance and detection results are the worst. Although the weight (108 MB) and the parameter amount (2.83×10^7) are lower than the YOLO models, the two-stage steps are complex and lead to the calculation amount (1.79×10^{11}) and the average detection time per frame (6.167 s) greatly exceed the YOLO models. The YOLOv3 model and YOLOv4 model still maintain better real-time detection results, and other indicators in the weight are close to the same, but the detection performance of the YOLOv4 model is better than the YOLOv3 model, where the F1 is 4.60% higher, mAP is 2.87% higher, Precision is 2.41% higher, and Recall is 6.54% higher especially. The EfficientNet-B0-YOLOv4 model proposed in this paper is slightly better than the YOLOv4 model in detection performance, where the F1 is 0.18% higher, mAP is 1.30% higher, Precision is 2.70% lower, and Recall is 2.86% higher especially. But in terms of weight indicators, the improved model is much better than the YOLOv4 model, where the average detection time per frame is reduced by 0.072 s, the weight size is reduced by 86 MB, the parameter amount is reduced by 2.62×10^7 , and the calculation amount is reduced by 1.72×10^{10} .

It can be seen from Figure 11 that the P-R curve area under the EfficientNet-B0-YOLOv4 model proposed in this paper is larger, which shows that it has better performance. It can be seen from Table 8 and Figure 12 that in the case of large objects, each model can accurately detect apples, but in the case of small objects, especially in the case of occlusion, the Faster R-CNN with ResNet model will have more missed detections and false detections, which leads to low Precision (52.07%). At the same time, the YOLO models will have

fewer missed detections and false detections, the detection result of the YOLOv3 model is relatively poor, the detection result of the YOLOv4 model and the EfficientNet-B0-YOLOv4 model proposed in this paper are close to the same.

Based on the above analysis, the whole results of the EfficientNet-B0-YOLOv4 model proposed in this paper are better than the current popular apple detection models, which can achieve high-recall and real-time detection performance, and reduce the weight size and computational complexity. The experimental results show that the proposed method in this paper is well applied to the vision system of the picking robot.

5. Conclusions

To simulate the possible complex scenes of apple detection in orchards and improve the apple dataset, an illustration data augmentation method is proposed and 8 common data augmentation methods are utilized to expand the dataset. On the expanded 2670 samples, the F1 of using the illustration data augmentation method has increased the most. Given the large size and computational complexity of the YOLOv4 model, EfficientNet is utilized to replace its backbone network CSPDarknet53. The improved EfficientNet-B0-YOLOv4 model has the F1 of 96.54%, the mAP of 98.15%, the Recall of 97.43%, and the average calculation time per frame of 0.338 s, which are better than the current popular YOLOv3 model, YOLOv4 model, and Faster R-CNN with ResNet model. Comparing the proposed EfficientNet-B0-YOLOv4 model with the original YOLOv4 model, the weight size is reduced by 35.25%, the parameter amount is reduced by 40.94%, and the calculation amount is reduced by 57.53%. In future work, we hope to add more apple classes for detection, and conduct level evaluation for each class after picking. For example, each class is divided into three levels: good, medium, and bad, thus forming a complete set of the apple detection system. Furthermore, we will continue to consolidate the illustration data augmentation method to improve the dataset.

Author Contributions: Conceptualization, L.W. and J.M.; Funding acquisition, J.M. and Y.Z.; Investigation, J.M. and H.L.; Supervision, L.W., J.M., Y.Z. and H.L.; Writing—original draft, L.W.; Writing—review & editing, L.W., J.M. and Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Hebei Natural Science Foundation (No. F2020202045), the Hebei Postgraduate Innovation Funding Project (No. CXZZBS2020026).

Institutional Review Board Statement: The studies not involving humans or animals.

Informed Consent Statement: All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: The raw data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

Acknowledgments: This work has been supported by Hebei Natural Science Foundation (Grant No. F2020202045) and Hebei Postgraduate Innovation Funding Project (Grant No. CXZZBS2020026).

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

References

1. Zhu, G.; Tian, C. Determining sugar content and firmness of ‘Fuji’ apples by using portable near-infrared spectrometer and diffuse transmittance spectroscopy. *J. Food Process Eng.* **2018**, *41*, e12810. [[CrossRef](#)]
2. Lehnert, C.; Sa, I.; McCool, C.; Upcroft, B.; Perez, T. Sweet pepper pose detection and grasping for automated crop harvesting. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA’16), Stockholm, Sweden, 16–21 May 2016; pp. 2428–2434.
3. Valdez, P. Apple Defect Detection Using Deep Learning Based Object Detection for Better Post Harvest Handling. *arXiv* **2020**, arXiv:2005.06089.
4. Cao, Y.; Qi, W.; Li, X.; Li, Z. Research progress and prospect on non-destructive detection and quality grading technology of apple. *Smart Agric.* **2019**, *1*, 29–45.

5. Zhang, J.; Karkee, M.; Zhang, Q.; Zhang, X.; Majeed, Y.; Fu, L.; Wang, S. Multi-class object detection using faster R-CNN and estimation of shaking locations for automated shake-and-catch apple harvesting. *Comput. Electron. Agric.* **2020**, *173*, 105384. [\[CrossRef\]](#)
6. Tian, Y.; Yang, G.; Wang, Z.; Li, E.; Liang, Z. Detection of apple lesions in orchards based on deep learning methods of cyclegan and yolov3-dense. *J. Sens.* **2019**, *2019*, 7630926. [\[CrossRef\]](#)
7. Mureşan, H.; Oltean, M. Fruit recognition from images using deep learning. *Acta Univ. Sapientiae Inform.* **2018**, *10*, 26–42. [\[CrossRef\]](#)
8. Tian, Y.; Yang, G.; Wang, Z.; Wang, H.; Li, E.; Liang, Z. Apple detection during different growth stages in orchards using the improved YOLO-V3 network. *Comput. Electron. Agric.* **2019**, *157*, 417–426. [\[CrossRef\]](#)
9. Kang, H.; Chen, C. Fast implementation of real-time fruit detection in apple orchards using deep learning. *Comput. Electron. Agric.* **2020**, *168*, 105108. [\[CrossRef\]](#)
10. Mazzia, V.; Khaliq, A.; Salvetti, F.; Chiaberge, M. Real-Time Apple Detection System Using Embedded Systems with Hardware Accelerators: An Edge AI Application. *IEEE Access* **2020**, *8*, 9102–9114. [\[CrossRef\]](#)
11. Kuznetsova, A.; Maleva, T.; Soloviev, V. Using YOLOv3 Algorithm with Pre- and Post-Processing for Apple Detection in Fruit-Harvesting Robot. *Agronomy* **2020**, *10*, 1016. [\[CrossRef\]](#)
12. Gao, F.; Fu, L.; 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. *Comput. Electron. Agric.* **2020**, *176*, 105634. [\[CrossRef\]](#)
13. Liu, X.; Zhao, D.; Jia, W.; Ji, W.; Sun, Y. A detection method for apple fruits based on color and shape features. *IEEE Access* **2019**, *7*, 67923–67933. [\[CrossRef\]](#)
14. Jia, W.; Tian, Y.; Luo, R.; Zhang, Z.; Lian, J.; Zheng, Y. Detection and segmentation of overlapped fruits based on optimized mask R-CNN application in apple harvesting robot. *Comput. Electron. Agric.* **2020**, *172*, 105380. [\[CrossRef\]](#)
15. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16), Las Vegas, NV, USA, 27 June 2016; pp. 779–788.
16. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17), Hawaii Convention Center, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
17. Redmon, J.; Farhadi, A. YoloV3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
18. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
19. Tan, M.; Le, Q.V. Efficientnet: Rethinking network scaling for convolutional neural networks. *arXiv* **2019**, arXiv:1905.11946.
20. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv* **2017**, arXiv:1710.09412.
21. Yun, S.; Han, D.; Oh, S.J.; Chun, S.; Choe, J.; Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV'19), Seoul, Korea, 27 October–3 November 2019; pp. 6023–6032.
22. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A new backbone that can enhance learning capability of cnn. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'20), Seattle, WA, USA, 14–19 June 2020; pp. 390–391.
23. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [\[CrossRef\]](#)
24. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17), Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
25. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18), Salt Lake City, UT, USA, 18–22 June 2018; pp. 8759–8768.
26. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [\[CrossRef\]](#)
27. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [\[CrossRef\]](#)
28. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
29. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16), Las Vegas, NV, USA, 27 June 2016; pp. 770–778.
30. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15), Boston, MA, USA, 7–22 June 2015; pp. 1–9.
31. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
32. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7132–7141.
33. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IOU Loss: Faster and Better Learning for Bounding Box Regression. In Proceedings of the 2020 AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 12993–13000.