2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC) | 978-1-6654-8810-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/COMPSAC54236.2022.00056

Apple-YOLO: A Novel Mobile Terminal Detector Based on YOLOv5 for Early Apple Leaf Diseases

Jinjiang Li
College of Information Engineering
Northwest A&F University
Yangling, China
2019_ljj@nwafu.edu.cn

Xianyu Zhu
College of Information Engineering
Northwest A&F University
Yangling, China
zhuxy@nwafu.edu.cn

Runchang Jia
College of Information Engineering
Northwest A&F University
Yangling, China
jiarunchang@nwafu.edu.cn

Bin Liu*
College of Information Engineering
Northwest A&F University
Yangling, China
liubin0929@nwsuaf.edu.cn

Cong Yu
College of Information Engineering
Northwest A&F University
Yangling, China
yc@nwafu.edu.cn

Abstract—Early detection of apple leaf diseases is the basis for timely precautions, which can inhibit the spread of the diseases and minimize the severe economic loss. Nowadays, CNN-based models are used for apple leaf diseases detection. However, due to the large model size and inference delay, the model is challenging to be transplanted to mobile terminals with good detection performance. This paper proposes a lightweight detection model Apple-YOLO on mobile terminals for real-time apple leaf diseases detection. First, a dataset named AppleSet8 is constructed using digital image processing and Mosaic data augmentation to improve the robustness and generalization ability of the model. Then the double-branch Apple-CSP module is presented to reduce the model parameters and guarantee feature extraction capability. Furthermore, the improved FDSA (Focus layer with depthwise separable convolution and attention mechanism) module effectively decreases the model's FLOPs and enhances the network's attention to the disease spots. Finally, the Skip-Spp (Skip-connection and Spatial pyramid pooling) module is built to strengthen the detection performance for multi-scale disease spots. The experiment results show that mobile-based Apple-YOLO has achieved 96.04% mAP, the inference speed of 34 FPS, and the size is only 5.33 MB, indicating that Apple-YOLO is suitable for the real-time detection of early apple leaf diseases in the real scenario.

Index Terms—apple leaf diseases, mobile terminal detection, convolutional neural networks, lightweight model

I. Introduction

China is the world's largest producer and consumer of apples, accounting for more than 40% of the world's total apple planting area and occupying an important position in the world apple industry. Due to the influence of climate, apple leaves are inevitably infected by different diseases, which will significantly affect the quality and yield of apples and cause substantial economic loss. Therefore, the most urgent issue at hand is to detect and control apple leaf diseases in a timely and effective manner [17].

The traditional way to detect crop diseases relies heavily on experts' experience to judge. However, this takes a lot of labor and time, and accuracy is also not guaranteed. In recent years, due to the accelerated process of agricultural informatization, machine learning-based algorithms [6], [24] have been applied to the agriculture field. However, machine learning algorithms, such as Support Vector Machine [18], Random Forest [7] and Decision Tree [2], require professional experience and manual feature selection, which limits their application in practical scenarios. Furthermore, machine learning-based models cannot achieve satisfactory accuracy because they are susceptible to selected features. With the development of deep learning, deep convolutional neural network is used to monitor crop diseases [4], [29]. The detection of crop diseases based on the deep convolutional neural network can automatically extract the features of different diseases and reach high accuracy, improving detection stability and providing an efficient end-to-end method for early apple leaf diseases detection. However, most CNN-based models contain many parameters and are computationally complex, with the training process relying heavily on high-performance servers. Meanwhile, the trained model is also difficult to be deployed to resource-constrained mobile terminals due to its large size.

To address the problems of the above researches, this paper proposes a novel lightweight detection model Apple-YOLO with high accuracy and fast inference time. The main contributions of this paper are listed below:

- This paper establishes a dataset AppleSet8 to enhance the generalization ability and robustness of the detection model, containing eight common early

Shaanxi's Key Research and Development Program (2021NY-138), by CCF-Baidu Open Fund(NO. 2021PP15002000) and China's National College Innovation and Entrepreneurship Training Program of Northwest A&F University (No.S202110712607).

*Bin Liu is the corresponding author (e-mail: liubin0929@nwsuaf.edu.cn)

apple leaf diseases. First, traditional digital image processing is used to expand the dataset to provide sufficient samples for model training. Also, a novel Mosaic data augmentation method is applied to enrich image background information and reduce the performance demand on GPU.

- A novel lightweight detection method on mobile terminals is constructed. To reduce the parameters, following the structural advantages of the original CSPNet, the brand new double-branch module Apple-CSP replaces the standard convolution and multi-ResBlock by using depthwise separable convolution and Ghost-BottleNeck. What is more, an FDSA module in AppleNet based on slicing operation and CBAM attention mechanism is proposed for decreasing the FLOPs of the model.
- Feature pyramid-based strategy enhances the detection performance of multi-scale apple leaf diseases. First, the Skip-Spp module is constructed by fusing shallow and deep features and parallelizing multiple max-pooling layers with different kernel sizes for extracting multi-scale disease spot features. In addition, two feature pyramid networks, FPN, and PANet enhance the location and semantic information of features respectively by fusing feature maps of different depths.

The rest of the paper is organized as follows: Section II describes the related work. Section III presents the overall framework of Apple-YOLO and the core modules. Section IV performs the experiments and gives a detailed performance analysis. Section V summarizes paper in a nutshell.

II. Related work

In the agricultural field, plant diseases are a major threat to plant growth and yield. Researchers are constantly trying to find better ways to control these diseases and have made outstanding contributions.

Traditional machine learning algorithms are first transferred to the agricultural field to detect plant disease. In [23], Sethy reported a novel approach for rice leaf diseases by using Gray Level Co-occurrence matrix for feature extraction and using SVM classifier for disease classification. The detection performance of the planned methodology got an accuracy of 97.91%. In [27], Sun proposed a new algorithm combining SLIC (Simple Linear Iterative Cluster) with SVM. Super-pixel block was obtained by SLIC algorithm, and classification map was obtained by classifying the super-pixel blocks with the help of SVM classifier. The accuracy, precision, recall rate, and F value on 261 diseased images were 98.5%, 96.8%, 98.6%, and 97.7%, respectively. However, machine learning algorithms require complex image pre-processing to obtain high-quality images and expertise in extracting disease features, not guaranteeing detection accuracy.

Convolutional neural networks have developed rapidly in recent years and have achieved good results in agriculture. In [14], Kiwi-ConvNet was built with Kiwi-Inception structures and dense connectivity strategy, which can enhance the capability of multi-scale feature extraction and ensure multi-dimensional feature fusion. In [26], Sun proposed a new detection model, named MEAN-SSD, by using MEAN block and Apple-Inception. The improved MEAN-SSD reached the detection speed of 12.53 FPS and 83.12% mAP. In [32], Xie proposed a faster DR-IACNN model with higher feature extraction capability by introducing the Inception and Inception-ResNet module. The mAP can reach 81.1%, but the speed is only 15.01 FPS. The above studies show that CNN-based models have been widely used for crop disease identification and detection with good results. However, the above detection models are still not accurate enough to detect confusing early apple leaf diseases and lack a lightweight model deployed to mobile for real-time detection. Therefore, a high-performance real-time detection model for early apple leaf diseases that can be deployed to mobile is proposed through our research.

III. Lightweight early apple leaf disease detection model

This section describes in detail the detection model Apple-YOLO. First section III-A introduces the construction of AppleSet8. The improved FDSA module, double-branch Apple-CSP module and Skip-Spp module are presented in section III-B, section III-C, section III-D, respectively. Finally, the overall framework of Apple-YOLO is described in section III-E.

A. Construction of the dataset AppleSet8

Fig. 1 illustrates the overall process of dataset construction and the important role played by the dataset.

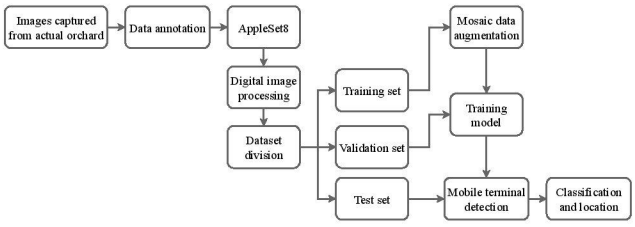


Fig. 1. Dataset generation and augmentation

1) Collection of original images: The original images of early apple leaf diseases are collected during April and May in the apple planting base in Qian County, Shaanxi Province, China. A total of 1,587 original images of early apple leaf diseases are obtained, containing eight common categories: Aphid, Alternaria blotch, Brown spot, Leaf miner, Mosaic, Powdery mildew, Spider mite, and Rust and Fig. 2 illustrates the disease features of eight different early apple leaf diseases. Also, to meet the demand for sufficient training samples for deep learning-based networks, the original 1,587 images are expanded

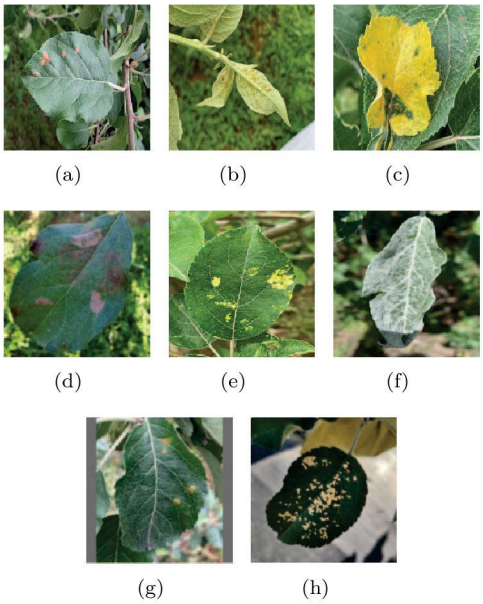


Fig. 2. Images of eight common types of early apple leaf diseases. (a) Alternaria blotch. (b) Aphid. (c) Brown spot. (d) Leaf miner. (e) Mosaic. (f) Powdery mildew.(g) Rust. (h) Spider mite.

to 24,757 images by traditional digital image processing, such as rotating, flipping, cropping, and scaling.

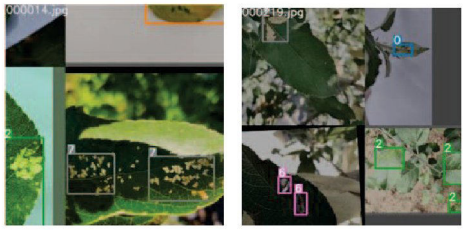


Fig. 3. Mosaic data augmentation

2) Mosaic data Augmentation: In recent years, some novel data enhancement methods such as Cutout [3] and Cutmix [33] strategies have been applied to object detection and achieved good performance. Unlike traditional data enhancement methods that process one image at a time, novel data enhancement methods process multiple images simultaneously and integrate them into a single image, improving the generalization and robustness of the model more effectively. So, improved Mosaic data augmentation based on Cutmix is used to further enhance the model's detection performance and reduce the dependence of model training on server performance. As Fig. 3 shows that Mosaic data augmentation randomly selects four images, scaling them and then stitching them together to form a complete image. The stitching of one image incorporates the background information of four different images and the disease spots, which moderates the problem of the large gap between positive and negative samples. What is more, more small disease spots are added to the images after performing random scaling, which is helpful to detect diseases with small spots. Last but not least, the data of four images is computed at once during batch normalization, which allows the model to have strong generalization ability without setting a large mini-batch.

B. The FDSA module

The current detection model's backbone network is accustomed to realizing the downsampling of input images by using multiple convolutions. However, using standard 3×3 convolution with the stride of 2 for downsampling will increase the computational complexity and FLOPs of the model compared to max-pooling. As shown in Fig. 4, a brand-new FDSA module is proposed to reduce the FLOPs of the model while reducing the image information loss caused by downsampling, which replaces two standard convolutional layers and a CSP-BottleNeck module.

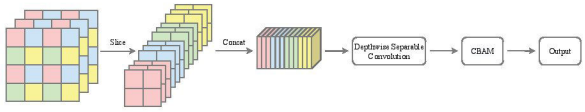


Fig. 4. FDSA module

The FDSA module first slices the input images, taking a value at every other pixel in the image. After the slicing operation, the height and width of the input images are reduced to half of the original images, and the number of channels is expanded to four times that of the original images. The slice operation concentrates the information on the height and width of the images into the channel space without loss of image information and floating-point operations.

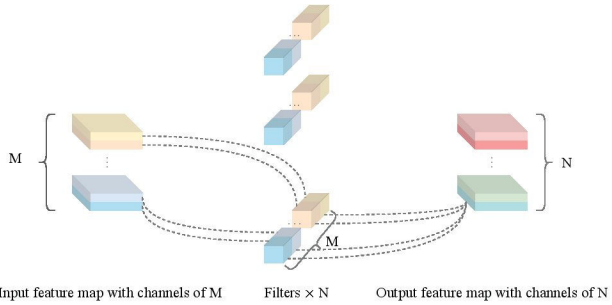


Fig. 5. Depthwise separable convolution

In order to adjust the number of channels of the output feature maps and reduce convolution operation's parameters, 3×3 depthwise separable convolution is used after the slicing operation. Different from standard convolution, as Fig. 5, shows the depthwise separable convolution consists of two parts: depthwise convolution and pointwise convolution. For depthwise convolution, one

TABLE I
Composition of AppleSet8

Disease	Total(original)	Total(augmented)	Training	Validation	Test
Alternaria blotch	131	2043	1363	340	340
Aphid	117	1825	1217	304	304
Brown spot	38	593	395	99	99
Leaf miner	360	5616	3744	936	936
Mosaic	183	2855	1903	476	476
Powdery mildew	94	1466	978	244	244
Rust	191	2980	1986	497	497
Spider mite	473	7379	4919	1230	1230
Total	1587	24757	16505	4126	4126

convolution kernel is only responsible for the convolution operation of one channel, so the channel number of the feature map will not be changed. Depthwise convolution makes the image features at the same position in different channels underutilized. Therefore, the pointwise convolution using multiple 1×1 convolution kernels is added to achieve the feature fusion among different channels and the channel expansion of the feature maps. Furthermore, considering that the background occupies most of the image and contains much noise, the model needs to pay more attention to the apple leaf spots. Hence, the CBAM module [31] is placed at the end of the FDSA module, along with the space and channel adaptively adjusting the feature maps.

C. double-branch Apple-CSP module

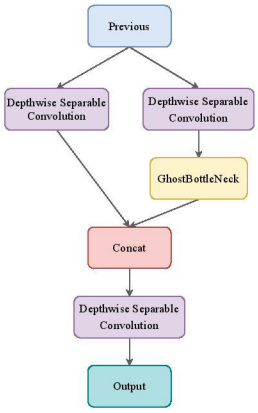


Fig. 6. Apple-CSP

Since the number of model parameters determines the model's size, it is essential to reduce the number of model parameters as much as possible for real-time detection on the mobile terminal. In addition, early apple leaf diseases show more minor spots and are easily confused, so it is also necessary to ensure that the model has strong feature extraction capabilities. Thus, as shown in Fig. 6, the Apple-CSP module is proposed inspired by CSPNet (Cross Stage Partial Network) [30] and Ghost-BottleNeck.

Following the structural advantages of the double-branch CSPNet, the improved Apple-CSP module effectively reduces the possibility of feature duplication in the feature fusion process by using splitting and concatenating strategies across stages. Compared to the original CSPNet, the feature maps in Apple-CSP are not split the number of channels in half and then divided into different branches. However, both branches achieve half the number of channels through convolution with a stride of 2, enhancing the effect of feature fusion.

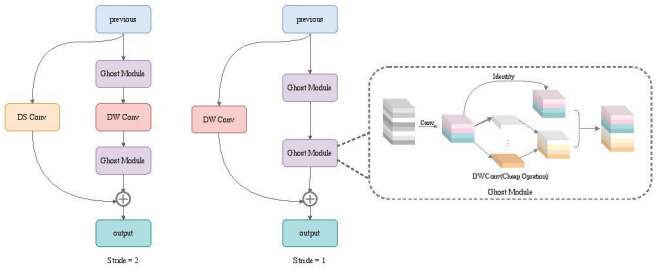


Fig. 7. Ghost-BottleNeck

Apple-CSP performs feature extraction by stacking multiple ResBlocks in one of the branches. However, it introduces many parameters though 1×1 convolution is used in ResBlock to reduce the number of model parameters. So ResBlock is replaced with Ghost-BottleNeck with a stride of 1 in the Apple-CSP module, and Fig. 7 shows the details of Ghost-BottleNeck. Ghost Module in Ghost-BottleNeck divides the input feature maps into two parts, one using standard convolution and the other using depthwise convolution. The output obtained after the two parts are concatenated with fewer parameters and ensures the quality of extracted features. To reduce the model parameters further, depthwise separable convolutions replace all standard convolutions in Apple-CSP except in Ghost Module.

D. The Skip-Spp module

The size of the area infected by different apple leaf diseases varies, and the size of the disease spots of the same disease is still not fixed. For Rust and Alternaria blotch, diseases with clusters of small spots, the disease

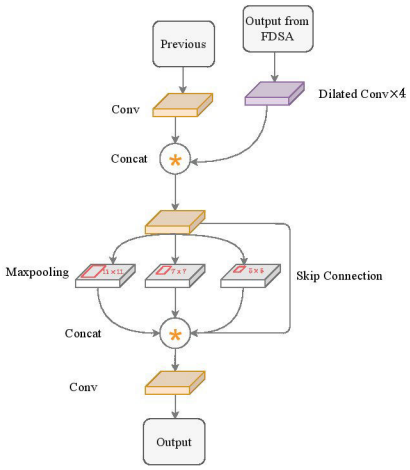


Fig. 8. Skip-Spp

spots of Rust are usually oily orange and protrude from the leaf surface. In contrast, disease spots Alternaria blotch are tan and wilt at the disease spots. For Leaf miner, the disease spots are gray and have a grid shape inside the disease spots. In the case of Aphids, the aphids are continuously distributed in clusters and concentrated on the tender leaves. For Spider mite, the disease spots are also checkered but light yellow and with larger infected areas. For diseases that infect the whole leaf, Mosaic-infected leaves are covered with bright yellow spots, Brown spot-infected leaves are yellow, and Powdery mildew-infected leaves are covered with a layer of white powder, and the leaves become curled.

Thus, The Skip-Spp module as shown in Fig. 8 is proposed to enhance the detection performance for the multi-scale disease spots by introducing the SPP-Net(Spatial Pyramid Pooling) [9]. The Skip-Spp module concatenates the output of the FDSA module through skip-connection, which can make better use of context information by fusing the shallow detail features and deep abstract features. Most importantly, the Skip-Spp module is divided into four branches: the max-pooling layer with kernel sizes of 5×5, 7×7, and 11×11 and other skip-connection. The max-pooling of different kernel sizes increases the receptive fields of the feature maps for the input images without increasing redundant parameters. Moreover, the fusion of local and global information enhances the model's ability to detect multi-scale disease spots.

E. Detection model Apple-YOLO

1) Feature extraction network AppleNet: In the course of CNN development, network models such as VGG [25], ResNet [10] and GoogLeNet [28] rely on high-performance servers to train models and achieve high accuracy, while the vast model size is their biggest drawback. In order to transplant and apply models to the mobile terminal, a large number of lightweight network models, such as ShuffleNet [34], MobileNet [11] and SqueezeNet [5], have reduced the size of the model while ensuring accuracy. Through experiments, it has been proved that GhostNet [8] is the best lightweight model for early apple leaf disease detection, and inspired by the structure of CSPNet with strong feature extraction ability, AppleNet is built based on these two models.

TABLE II
The related parameters of AppleNet

Name	Kernel Size/Stride	Output Size
FDSA	$3 \times 3/1$	$208 \times 208 \times 32$
Ghost-BottleNeck1	$3 \times 3/2$	$104 \times 104 \times 64$
Apple-CSP1	$3 \times 3/1$	$104 \times 104 \times 64$
Ghost-BottleNeck2	$3 \times 3/2$	$52 \times 52 \times 128$
Apple-CSP2	$3 \times 3/1$	$52 \times 52 \times 128$
Ghost-BottleNeck3	$3 \times 3/2$	$26 \times 26 \times 256$
Apple-CSP3	$3 \times 3/1$	$26 \times 26 \times 256$
Ghost-BottleNeck4	$3 \times 3/2$	$13 \times 13 \times 512$
GAP	——	$1 \times 1 \times 512$
FC[1]	——	$1 \times 1 \times 8$

[1] The dropout is set to 0.5

Table II shows the overall structure of AppleNet. The first layer of the model is an FDSA module that replaces two convolutional layers and a CSP-BottleNeck module, which reduces the parameters and FLOPs of the model while reducing feature loss. After that, the main feature extraction part of AppleNet is formed by stacking multiple sets of Ghost-BottleNeck and Apple-CSP modules. The Ghost-BottleNeck block with a step of 2 is used to downsample the feature maps, and the double-branch Apple-CSP ensures the ability to extract disease features while significantly reducing the model parameters. At the end of the backbone, a GAP(Global Average Pooling) layer is followed by an FC layer to get an 8-dimensional output to classify eight different early apple leaf diseases.

2) The feature pyramid: Although AppleNet can extract the main features of early apple leaf diseases, the model cannot extract the features of multi-scale spots due to convolutional kernels of the same size. In addition, detailed features are not fused with abstract features, so the network cannot adequately combine shallow location information and deep semantic information and utilize them. Hence it is necessary to fuse the feature maps carrying different information.

In the feature pyramid part, the Skip-Spp module is first used to enhance the model's detection effect on multi-scale diseases by parallelizing multiple max-pooling layers with different kernel sizes. Then, two feature pyramid networks, FPN (Feature Pyramid Networks) [13] and PANet (Path Aggregation Network) [15], are used to strengthen extracted features. The FPN structure uses a top-down feature fusion method that conveys robust semantic information from top to bottom. In contrast, PANet uses a bottom-up feature fusion approach to convey powerful localization information from bottom to top. So,

upsampled feature maps in FPN are fused with shallow feature maps in AppleNet, downsampled feature maps in PANet are fused with feature maps output in FPN.

3) Construction of Apple-YOLO: Based on all the above-improved modules, a mobile apple early leaf disease detection model is built, and Fig. 9 shows the overall framework of Apple-YOLO. Apple-YOLO's feature extraction network, AppleNet, significantly reduces model parameters while ensuring model detection accuracy. Then the feature pyramid structure lying after AppleNet fuses the feature maps of different stages and enhances the use of the model for different features. Finally, after adjusting the size of feature maps, three different sizes of prediction layers, 20×20×39, 40×40×39 and 80×80×39, are obtained for localization and classification of the small, medium, and large disease spots respectively.

IV. Experimental design and analysis

This section describes the experiment's setup and gives a detailed analysis of the experimental results.

A. Experimental setup

All experiments were carried out on an Ubuntu-16.04.2 server with an Intel Xeon E5-2678 v3. The high-performance server contains an NVIDIA RTX3090 with a 24 GB memory size used for model training. The specific environment of the high-performance server is shown in Table III. In addition, the Pytorch framework is used to build the Apple-YOLO model on a high-performance server.

TABLE III
High Performance Server Configuration

Configuration	Value
Central processing unit	Intel Xeon E5-2678 v3
Graphics processor unit	NVIDIA GeForce RTX 3090
Operation system	Ubuntu 16.04.2 LTS (64-bit)
Memory	122GB System memory
Language	Python 3.8
Deep learning framework	Pytorch 1.9.0

B. Training strategy

TABLE IV
The details of training strategy

Configuration	Value
Optimizer	SGD
Base Learning Rate	0.01
Momentum	0.937
Weight_decay	0.0005
Warmup_epochs	3
Warmup_momentum	0.8
Warmup_bias_lr	0.1
Training Epoch	120

Although the network structure design determines the upper limit of the model, the model's training process directly affects the model's performance. Therefore, a correct and effective training optimization strategy is required to obtain the optimal model.

In order to accelerate the convergence of the model during the training process and reduce the requirement for gradients in the backpropagation process, the SGD optimizer is selected. Moreover, the training strategy of Warmup is used to enhance the stability of the network during the early stages of training, where a lower learning rate is used to gradually increase to a larger learning rate when the model first starts training. Then the predetermined learning rate is used after the model is stabilized. The details of the training strategy are shown Table IV.

C. Accuracy comparison of different backbone networks

Apple-YOLO's backbone extracts the main features of early apple leaf diseases, which plays a vital role in the subsequent feature fusion and model prediction. In this section, the different types of the backbone are trained and tested on the AppleSet8 and compared with AppleNet.

Table V shows the performance metrics of the different backbone selected. For large backbones such as AlexNet [12], ResNet, and VGG, good performance has been achieved in recognition accuracy. However, the large size makes it challenging to transplant the model to mobile terminals. For lightweight backbones such as MobileNet, ShuffleNet, and SqueezeNet, the models' parameters and FLOPs decreased significantly. At the same time, the recognition accuracy exceeded that of large networks, indicating that a shallow and lightweight network structure can extract good early apple leaf disease features and classify them accurately. The lightweight GhostNet gets excellent performance in recognition accuracy and model parameters and FLOPs, which inspires us to make the most of their strengths to build AppleNet. After further compression of the model, the final AppleNet is optimal in size and parameters while maintaining a high recognition accuracy.

D. Apple-YOLO's classification performance of different diseases

Detection models may be confused when faced with easily confused disease spots, while the same diseases are different, making the classification more difficult. A confusion matrix after normalization is used to visually evaluate the classification performance of Apple-YOLO on eight kinds of early apple leaf diseases. Fig. 10 shows the confusion matrix of the final test results. It can be seen that the model has the best classification performance on the Brown spot, and all samples are predicted correctly. The detection performance remains high for Leaf miner, Spider mite, Powdery mildew, and other apple leaf diseases with big disease spots, and more than 90% of the samples are predicted correctly.

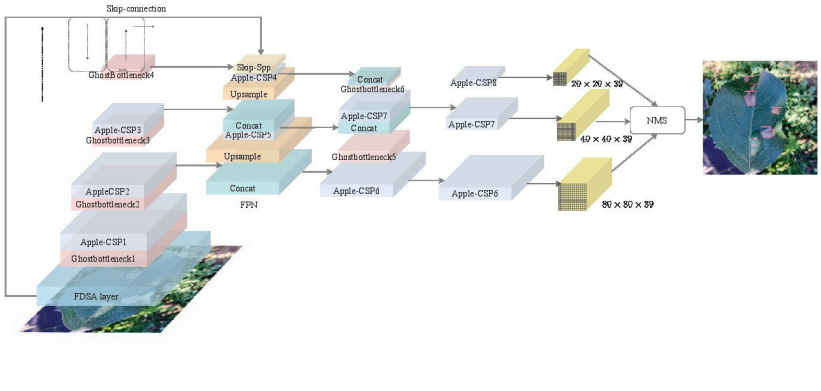


Fig. 9. Apple-YOLO

TABLE V
The performance of pre-network

Model	Accuracy(%)	Size(MB)	FLOPs(G)	Parameter(M)
AlexNet	94.78	55.69	0.309	14.6
VGG19	97.02	532.55	19.66	139.6
ResNet101	99.20	81.31	3.670	21.3
ResNeXt50	99.32	88.04	4.260	23
EfficientV2	99.23	77.76	2.870	20.18
MobileNetV3-small	97.62	6.67	0.305	4
SqueezeNet	96.88	2.83	0.734	0.74
ShuffleNetV2	97.47	4.94	0.147	1.26
Xception	99.37	79.69	4.580	20.8
GhostNet	99.20	15.12	0.149	3.9
AppleNet	98.78	1.61	0.386	0.389

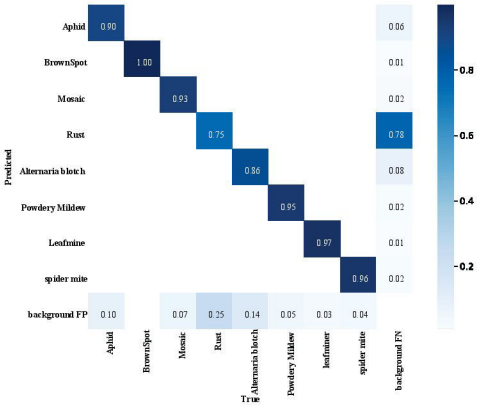


Fig. 10. Confusion matrix after normalization

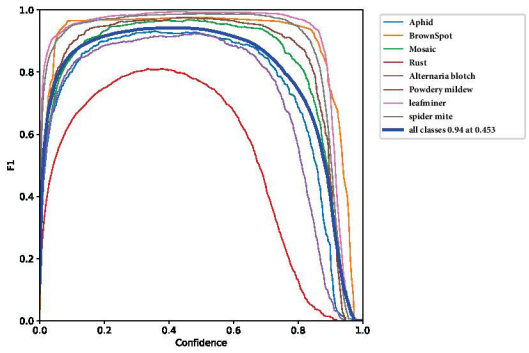


Fig. 11. $F1$ score

In contrast, the detection performance is poor for the two minor spot diseases, Rust and Alternaria blotch. Especially for Rust, only more than 70% of the samples are predicted correctly. The main reason for this result is that the early Rust spots are small, and the spot features are not apparent and easily clustered. Therefore, the model is difficult to extract and distinguish the features of early Rust correctly.

The $F1$ score curve for each disease is also given for a more comprehensive model evaluation. $F1$ score is the summed average of precision and recall, taking both into account to evaluate the model. Fig. 11 shows that the average $F1$ score of the model for all diseases is high when the confidence level is between 0.2 and 0.7, and the highest average $F1$ score of 0.94 is achieved at a confidence level of 0.453. The results show that Apple-YOLO can classify the eight early apple leaf diseases well and satisfy practical application scenarios' detection needs.

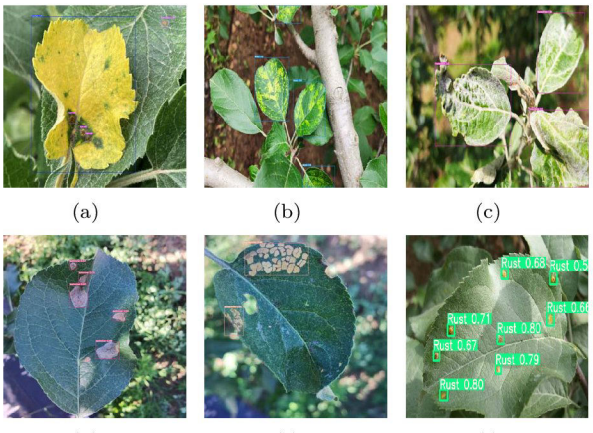
E. Performance comparison of Apple-YOLO on mobile terminals

In order to compare the detection performance of different object detection algorithms on AppleSet8, the typical one-stage object detection algorithms SSD [16] and YOLO [1], [19]–[21] and Faster-RCNN [22], a representative of two-stage detection algorithms, have been selected for the detection of early diseases of apple leaves. The mean Average Precision (mAP) is the most critical metric for evaluating detection models. In addition, since the model needs to be transplanted to mobile terminals, inference speed and model size are also used as essential evaluation metrics. The detection results of different algorithms are shown in Table VI.

For the two-stage object detection algorithm Faster-RCNN, detection accuracy is the most significant advantage. However, Faster-RCNN only reaches 72.48% mAP while RPN usage and two fully connected layers dramatically increase the model's parameters.

For other one-stage object detection YOLO algorithms, since YOLOv3, the feature extraction capability of the network has been enhanced by introducing the FPN structure to the model. Because the backbone DarkNet53 and CSP-DarkNet of YOLOv3 and YOLOv4 use multiple RseBlocks, the model size is bigger than Faster-RCNN. YOLOv5-s compressed the model simultaneously in both the width and depth of the model, so the model size is only 27.93 MB. Due to the advantages of the YOLOv5-s baseline design, the mAP reaches an amazing 98.53%.

The detection results of different apple leaf diseases in Fig. 12 show the practicality of Apple-YOLO. The mobile-oriented early apple leaf diseases detector Apple-YOLO further compresses the model size after using the proposed FDSA, Apple-CSP, and Skip-Spp module and maintains the detection mAP at 96.04%, which is still at a very competitive level. In addition, detection speeds up to 34 FPS, beyond the practical application requirements for mobile terminals.



(a) (b) (c)

(d) (e) (f)

Fig. 12. Early apple leaf diseases detection result

F. Ablation study

The Apple-YOLO uses Apple-CSP, FDSA, and Skip-Spp modules, and the Mosaic data augmentation method is adopted in model training. In order to reflect the influence of the above four different operations on the model, four experiments were carried out on AppleSet8 based on the original Apple-YOLO, and experimental results have been shown in Table VII.

The FDSA module uses slicing operations instead of downsampling to reduce the information loss of input images. The FDSA module that replaces two standard 3×3 convolutions and one CSP-BottleNeck module uses only one depthwise separable convolution, which makes the FLOPs of the model drop from 7.99G to 4.22G. Furthermore, with the addition of the CBAM attention module in the FDSA module, the mAP of the model is improved by 0.6 percentage points.

The Mosaic data augmentation method has dramatically improved the mAP of Apple-YOLO. After using the method, the mAP of the model increased by 5%. Mosaic data augmentation increases the background complexity by splicing four images together, which is equivalent to inputting four images at a time for training. The four images are scaled randomly to increase the number of small disease spots conducive to the model's detection for Rust and Alternaria blotch diseases while alleviating the problem of the large gap between positive and negative samples.

When the Skip-Spp module is not used, many parameters caused by the four dilated convolutions in the Skip connection are omitted, so the model size is reduced from 5.3 MB to 1.7 MB. However, without the shallow detail information brought by skip connection and the multi-scale information fused by the Spp layer, the mAP is reduced by four percentage points. Therefore, the Skip-Spp module can effectively improve the detection performance for multi-scale disease spots.

After using the Apple-CSP module, the size of Apple-YOLO has been reduced from 12.4 MB to 5.3 MB. In contrast, mAP has only dropped by 2.5%, but this is still at a high level and can meet the needs of accurate detection for early apple leaf diseases on mobile terminals. It shows that using depthwise separable convolution and Ghost-BottleNeck in the Apple-CSP module significantly reduces the parameters, and the CSP structure enhances the network's ability to extract features.

G. Mobile terminal model deployment

In order to verify the performance of Apple-YOLO in practical application, Apple-YOLO is transplanted from high-performance to mobile terminal devices. The way of transplantation is shown in Fig. 13. First, the model file in *pth* format is converted to the *pt* format, containing the model's information. Secondly, utilizing the TorchMobile tool, the *pt* file is optimized to the *ptl* file to adapt the resource-limited device. Then, the *apks* file is generated

TABLE VI
The performance of detection model

method	SSD	Faster-RCNN	YOLOv3	YOLOv4	YOLOv5-s	Apple-YOLO
backbone	ResNet50	ResNet50	DarkNet53	CSPDarkNet	CSPDarknet	AppleNet
mAP(%)	78.3	72.48	86.18	87.38	98.53	96.04
Size(MB)	106.9	108.40	235.13	244.44	27.93	5.33
Speed(FPS)	12	8	6	7	33	34

TABLE VII
The result of ablation study

FDSA	Apple-CSP	Mosaic	Skip-Spp	mAP(%)	Size(MB)	FLOPs(G)	Param(M)
✗	✓	✓	✓	95.42	5.3	4.22	2.45
✓	✗	✓	✓	98.58	12.4	7.99	6.19
✓	✓	✗	✓	90.44	5.3	4.22	2.45
✓	✓	✓	✗	92.11	1.7	1.21	0.67
✓	✓	✓	✓	96.04	5.3	4.22	2.45

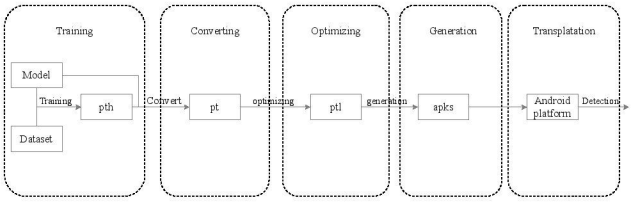


Fig. 13. Mobile terminal model deployment process

based on Android Studio. Finally, mobile terminals can realize real-time inference by detecting the image captured from the screen. The final interface of the app is shown in Fig. 14.

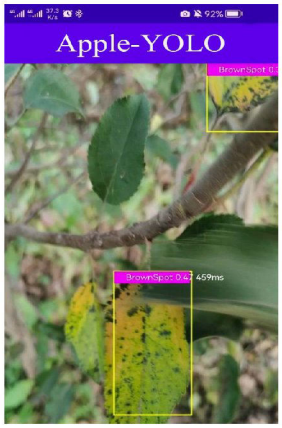


Fig. 14. Apple leaf diseases detection results on the mobile phone

V. Conclusions

This paper proposes a mobile-oriented early apple leaf disease detector Apple-YOLO, which can automatically extract features of different disease spots and be deployed to mobile terminals for accurate and real-time detection of eight different diseases. Traditional digital image processing provides sufficient images for AppleSet8, containing 24,757 early apple leaf diseases images. The Mosaic data augmentation method is used to ensure the generalization ability and robustness of the model. The proposed Apple-YOLO detector reduces the model's parameters and FLOPs using the Apple-CSP and FDSA modules. The Apple-YOLO's multi-scale disease spots detection performance is enhanced by using the Skip-Spp module.

A deep learning-based detection model is implemented in the Pytorch framework on the high-performance server. The detection performance of Apple-YOLO achieves 96.04% mAP, the detection speed of 34 FPS, and the model size is only 5.33 MB. More importantly, the model is successfully transplanted to a Huawei Mate 40 phone to detect the disease in real-time. The experimental results show that Apple-YOLO can detect eight common early apple leaf diseases accurately and quickly, which provides an innovative and practical method for lightweight model design and deployment.

Author contributions

Jinjiang Li took on the main work of proposing the innovation, designing and completing the main experiments, and writing the first draft of the paper. Xianyu Zhu was responsible for organizing the experimental data and visualizing the results. Bin Liu made a significant contribution of giving constructive advice, providing the experimental environment, financial support, and was responsible for reviewing and revising the draft. Runchang Jia and Cong Yu helped to revise the paper.

Acknowledgment

The author are grateful to our colleagues who have been accompanying us to make progress together.

References

[1] Bochkovskiy, Alexey and Wang, Chien-Yao and Liao, Hong-Yuan Mark. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2:7, 2020.

[2] Jayraj Chopda, Hiral Raveshiya, Sagar Nakum, and Vivek Nakrani. Cotton crop disease detection using decision tree classifier. In Proceedings of the International Conference on Smart City and Emerging Technology, pages 1–5, 2018.

[3] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. arXiv e-prints, pages arXiv–1708, 2017.

[4] Saraswathi Ea and Usha Kiruthikab. Study on stages of diseases detection in plant using deep convolutional neural network (cnn) in agriculture. Turkish Journal of Computer and Mathematics Education Vol, 12(9):2570–2576, 2021.

[5] N Iandola Forrest, Han Song, W Moskewicz Matthew, Ashraf Khalid, J Dally William, and K Kurt. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. In Proceedings of the International Conference on Learning Representations, pages 207–212, 2017.

[6] BV Gokulnath and G UshaDevi. A survey on plant disease prediction using machine learning and the deep learning techniques. Inteligencia Artificial, 23(65):136–154, 2020.

[7] Meghana Govardhan and MB Veena. Diagnosis of tomato plant diseases using random forest. In Proceedings of the Global Conference for Advancement in Technology, pages 1–5, 2019.

[8] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. Computing Research Repository, pages 1–8, 2019.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(9):1904–1916, 2015.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770–778, 2016.

[11] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. Computing Research Repository, 126:1–9, 2017.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25:1097–1105, 2012.

[13] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2117–2125, 2017.

[14] Bin Liu, Zefeng Ding, Yun Zhang, Dongjian He, and Jinrong He. Kiwifruit leaf disease identification using improved deep convolutional neural networks. In Proceedings of the IEEE 44th Annual Computers, Software, and Applications Conference, pages 1267–1272, 2020.

[15] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In Proceedings of the Conference on Computer Vision and Pattern Recognition, pages 8759–8768, 2018.

[16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer, 2016.

[17] Mohamed Loey, Ahmed ElSawy, and Mohamed Afify. Deep learning in plant diseases detection for agricultural crops: a survey. International Journal of Service Science, Management, Engineering, and Technology, 11(2):41–58, 2020.

[18] Kapil Prashar, Rajneesh Talwar, and Chander Kant. Robust automatic cotton crop disease recognition (acdr) method using the hybrid feature descriptor with svm. In Proceedings of the 4th 2016 International Conference on Computing on sustainable Global Development, pages 1–6, 2017.

[19] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 779–788, 2016.

[20] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7263–7271, 2017.

[21] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. arXiv e-prints, pages arXiv–1804, 2018.

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6):1137–1149, 2017.

[23] Prabira Kumar Sethy, Nalini Kanta Barpanda, and Amiya Kumar Rath. Detection and identification of rice leaf diseases using multiclass svm and particle swarm optimization technique. International Journal of Innovative Technology and Exploring Engineering, 8(6S2):108–120, 2019.

[24] Prabira Kumar Sethy, Nalini Kanta Barpanda, Amiya Kumar Rath, and Santi Kumari Behera. Image processing techniques for diagnosing rice plant disease: A survey. Procedia Computer Science, 167:516–530, 2020.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Proceedings of the International Conference on Learning Representations, pages 1–12, San Diego, CA, 2015. ICLR.

[26] Henan Sun, Haowei Xu, Bin Liu, Dongjian He, Jinrong He, Haixi Zhang, and Nan Geng. Mean-ssd: A novel real-time detector for apple leaf diseases using improved light-weight convolutional neural networks. Computers and Electronics in Agriculture, 189:1–11, 2021.

[27] Yunyun Sun, Zhaohui Jiang, Liping Zhang, Wei Dong, and Yuan Rao. Slic_svm based leaf diseases saliency map extraction of tea plant. Computers and electronics in agriculture, 157:102–109, 2019.

[28] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1–9, 2015.

[29] Muammer Turkoglu, Berrin Yanikoğlu, and Davut Hanbay. Plantdiseasenet: Convolutional neural network ensemble for plant disease and pest detection. Signal, Image and Video Processing, 9:1–9, 2021.

[30] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. Cspnet: A new backbone that can enhance learning capability of cnn. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pages 390–391, 2020.

[31] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision, pages 3–19, 2018.

[32] Xiaoyue Xie, Yuan Ma, Bin Liu, Jinrong He, Shuqin Li, and Hongyan Wang. A deep-learning-based real-time detector for grape leaf diseases using improved convolutional neural networks. Frontiers in plant science, 11:751–765, 2020.

[33] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 6023–6032, 2019.

[34] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 6848–6856, 2018.