IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 May 2023, accepted 30 May 2023, date of publication 2 June 2023, date of current version 7 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3282309

APPLIED RESEARCH

An Improved Lightweight YOLOv5 Algorithm for Detecting Strawberry Diseases

SHUNLONG CHEN, YINGHUA LIAO, FENG LIN, AND BO HUANG
School of Mechanical Engineering, Sichuan University of Science and Engineering, Yibin 644000, China

Corresponding author: Yinghua Liao (liaoyinghua118@163.com)

This work was supported in part by the Key Research and Development Project of Science and Technology Department of Sichuan Province under Grant 2021YFG0056; and in part by the Innovation Fund of Postgraduate, Sichuan University of Science and Engineering, under Grant Y2022046.

ABSTRACT This paper proposes an improved lightweight YOLOv5 model for the real-time detection of strawberry diseases. The ghost convolution (GhostConv) module is incorporated into the YOLOv5 network, reducing the parameter numbers and floating-point operations (FLOPs) for extracting feature information using the backbone network. An involution operator is utilized in the backbone network to expand the receptive field, enhance the spatial information on strawberry disease characteristics, and reduce the number of FLOPs in the model. A convolutional block attention module (CBAM) is incorporated into the backbone network to enhance the network's ability to extract strawberry disease features and suppress non-critical information. The upsampling module is replaced by a lightweight upsampling operator called Content-Aware ReAssembly of Features (CARAFE), which extracts feature map information and enhances the ability to focus on strawberry disease features. The experimental results on an open-source strawberry disease dataset show that the model achieves mean average precision (mAP)@0.5 of 94.7% with 3.9 M parameters and 3.6 G FLOPs. The improved model has higher detection precision than the original one and lower hardware requirements, providing a new strategy for strawberry disease identification and control.

INDEX TERMS Computer vision, image classification, lightweight network, YOLOv5.

I. INTRODUCTION

Strawberries have high nutritional value and economic value. The area of strawberry cultivation has increased as the market demand has risen. Strawberry diseases present a significant barrier to the extensive cultivation of strawberries. Bacteria, fungi, and pathogens are the primary causative agents of strawberry diseases, with these plant pathogens typically invading plants through their leaves, roots, and stems [1]. In the past, plant disease identification often relied on manual recognition and expert systems, but the efficiency and accuracy of diagnosis were significantly lower, hindering real-time crop monitoring. Early detection of plant diseases during the initial stage of infection is crucial for effective prevention, yet often proves challenging to promptly identify. It is necessary to identify diseases accurately and quickly during strawberry cultivation and implement correct and effective measures to curb disease spread, prevent yield and quality reduction, and minimize pesticide use [2]. Therefore, it is critical to obtain plant disease information in real time in smart agriculture. Traditional image recognition techniques have achieved good results but have limitations, such as complex image preprocessing, high subjectivity, and noise and interference in complex environments [3]. Due to technological advances in deep learning methods and computer hardware, deep learning-based target detection algorithms are increasingly used in agricultural research because of their high speed, precision, generalization ability, and robustness [4], [5].

The application of deep learning-based detection methods has greatly contributed to the detection and identification of plant diseases, effectively reducing the cost of manual diagnosis of plant diseases and providing valuable assistance to agricultural producers. However, strawberry diseases spread rapidly and through various paths, and negligence in management causes substantial yield reductions and affects

The associate editor coordinating the review of this manuscript and approving it for publication was Yizhang Jiang.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

strawberry quality. For example, strawberries can be infected with strawberry powdery mildew through the air from the seedling stage to the fruiting stage. It affects strawberry leaves, flowers, and fruits, and the disease spreads rapidly. Numerous spores are produced within 4-7 days after the outbreak of strawberry powdery mildew and infect other leaves through air currents [6]. It is necessary to identify diseases quickly and accurately and adopt appropriate control measures to avoid disease spread in strawberry fields. Although the current CNN-based plant disease detection method has achieved good results, the model has many parameters and high computational requirements. Embedded devices with limited computational performance and storage resources are not suitable for large-scale deployment. In this paper, an improved lightweight YOLOv5s model for strawberry disease detection is proposed to improve the automatic monitoring capability of strawberry diseases in greenhouses and provide guidance for disease control. This method detects strawberry diseases in complex natural environments. The main contributions of this study are as follows:

(1) We incorporate the ghost convolution (GhostConv) module into the YOLOv5 network to achieve model compression while ensuring high detection precision.
(2) The involution operator is used to assign weights adaptively at different spatial locations to extract a wide range of semantic information and substantially reduce the number of floating-point operators (FLOPs) of the model.
(3) The convolutional block attention module (CBAM) is added to the backbone network to improve its feature extraction capability and suppress non-critical information.
(4) The lightweight upsampling operator Content-Aware ReAssembly of Features (CARAFE) is used to replace the network's upsampling module, enabling the network to focus on the target region and enhancing the model's ability to model spatial information in complex natural environments.

The rest of this paper is organized as follows. Section II provides an overview of the application of traditional machine learning and deep learning-based detection techniques in the field of plant disease detection. Section III introduces the YOLOv5 model and the improved model. Section IV describes the experiments and results. Section V provides the discussion and the limitations of the methods and data. Section VI concludes the paper.

II. RELATED WORKS

In the domain of plant disease detection, numerous researchers have employed machine learning techniques to detect and classify plant diseases based on image data. The extraction and selection of disease features were designed based on prior knowledge and experience [7]. The detection performance of the algorithm primarily depends on whether the extracted and selected features can adequately represent the disease characteristics and the compatibility of the classifier. Additionally, it is influenced by factors such as plant types, disease types, and environmental lighting conditions. For example, Dubey and Jalal [8] employed the K-Means clustering algorithm to segment lesion regions, utilized the completed local binary pattern (CLBP) to extract color and texture features of apple diseases, and employed an improved support vector machine (SVM) for the detection of 93 types of apple diseases. The proposed solution algorithm achieved an accuracy rate of 93% in apple disease detection. Traditional image analysis techniques, combined with biological characteristics such as disease, texture, and shape of crops, enable the rapid identification of crop diseases. However, due to the complexity of plant disease symptoms, it is challenging for manual feature design optimization, leading to suboptimal performance in the recognition of plant diseases in complex scenarios.

Compared to early plant disease detection methods, many researchers have shifted their focus towards utilizing deep learning techniques for plant disease detection. This is due to the powerful feature representation capability, automatic feature learning, and robustness in complex natural environments exhibited by deep learning models. Deep learning demonstrates exceptional performance and adaptability, making it highly suitable for addressing the challenges associated with plant disease detection in diverse agricultural settings. For instance, Ferentinos [9] trained several convolutional neural networks (CNN) models for plant leaf disease detection. The Visual Geometry Group (VGG) CNN model obtained the highest detection accuracy of 99.53%, demonstrating the excellent performance of CNNs for detecting plant diseases. Anandhakrishnan and Jaisakthi [10] proposed an automatic system for identifying tomato leaf diseases based on deep CNNs. The model reduced the time required to identify tomato leaf diseases. Many deep-learning techniques have been applied to the detection and control of strawberry diseases. Nie et al. [11] proposed a faster R-CNN and multi-task learning-based network for detecting strawberry verticillium wilt. The network considered the polymorphic characteristics of the disease in the petioles and young leaves and automatically performed classification. Xiao et al. [12] used a CNN to train a ResNet50 model using five strawberry disease images. This model had the best detection performance and achieved 100% precision for the detection of leaf blight. Li et al. [13] studied strawberry powdery mildew and infected leaf detection in complex environments. They proposed a YOLOv4 network that used a deep convolution and hybrid attention mechanism, providing a solution for the early detection of strawberry powdery mildew in natural environments.

Deep learning models have been widely used in visual tasks such as image classification and object detection, achieving significant success. Deep learning typically involves training models with large and complex datasets, which can result in high training costs. Additionally, it requires high-performance hardware to execute the complex mathematical computations involved in model

inference [14]. The lightweighting of models is inherently interconnected with the detection of real-time capabilities, as lightweight models enable faster inference within limited resources and time constraints. This makes them suitable for scenarios with high demands for real-time performance, such as video analysis, facial recognition, and autonomous driving. Currently, model lightweighting focuses on reducing model parameters, minimizing computational complexity, and decreasing runtime. Researchers have proposed various methods for achieving model lightweighting, including directly constructing lightweight and compact network architectures, using knowledge distillation, low-rank factorization, and more.

Although these plant disease detection models have achieved impressive detection performance, they often come with a large number of parameters and computational requirements, resulting in high hardware costs when deployed on embedded terminals. Many scholars have conducted research on how to achieve real-time plant disease detection on mobile platforms with limited computing power. For instance, Shao et al. [15] proposed a lightweight convolutional neural network (L-CSMS) using channel shuffle operation and the multiple-size convolution module and applied it to an automatic plant disease severity identification and diagnosis system. The accuracy of the proposed lightweight model L-CSMS reached 90.6% and 97.9% on the plant disease severity dataset and PlantVillage dataset, respectively. Chen et al. [16] presented a lightweight network architecture, MobInc-Net, for crop disease detection. This architecture replaces the original convolutions in the MobileNet backbone network with Inception modules and adds a Softmax layer and SSD block after the foundation network. By applying two-stage transfer learning, MobInc-Net achieves an average accuracy of 99.21% on PlantVillage datasets. Fang et al. [17] proposed a lightweight plant disease classification model that incorporates the Grabcut, new coordinate attention, and channel pruning algorithms. The proposed model utilizes channel pruning algorithms to reduce both model size and computational complexity by 85.19% and 92.15%, respectively, enabling the network to meet the deployment requirements of low-storage and low-computational-power platforms. These lightweight neural network models have demonstrated excellent performance in recognizing plant leaf disease images with simple backgrounds. This is primarily due to their utilization of datasets created from images captured in controlled laboratory settings. These images often have relatively simple and similar backgrounds, resulting in models with limited robustness. Consequently, when these trained models are applied in real-world environments, their detection performance tends to decrease significantly.

In summary, these studies face two challenges in terms of guiding agricultural production. On one hand, there is a lack of experimental materials that are collected from real field environments. On the other hand, there is a need to strike a balance between the deployment cost of algorithms and detection performance. These are the pressing issues that need to be addressed to facilitate low-cost intelligent upgrades in agriculture.

III. PRINCIPLE OF THE DETECTION ALGORITHM

A. IMPROVEMENT OF THE YOLOv5 NETWORK ARCHITECTURE DESIGN

YOLOv5 [18] is a commonly used one-stage target detection algorithm, that benefits from the established PyTorch ecosystem which makes it simpler to support and easier to deploy. YOLOv5 was chosen for lightweighting improvements due to its unique combination of accuracy, efficiency, adaptability, and its significance in the computer vision community. It has five versions with different model widths and depths (n, s, m, l, x) and consists of an input, backbone network, neck network, and detection head. The input includes mosaic data enhancement, auto-learning bounding box anchors, and adaptive image scaling for preprocessing the input image. The backbone network consists of a Conv module, C3 [19] module, and a spatial pyramid pooling-fast (SPPF) module to extract image features. The neck network utilizes feature pyramid networks (FPNs) [20] and a path aggregation network (PAN) [21] to extract feature information from different-sized targets. The detection head is used to detect and classify the input image. Our objective is to design a lightweight target detection model with high detection performance and few numbers of FLOPs and parameters. Therefore, we selected the YOLOv5s model, which has good speed and precision as the baseline model, and improved it to reduce its hardware requirements. We present an enhanced, lightweight YOLOv5s model to achieve superior accuracy and speed in the detection of strawberry diseases. The architecture of this optimized model is depicted in Figure 1.

B. IMPROVED YOLOv5 NETWORK STRUCTURE

Traditional Convolutional Neural Networks (CNNs) attain the desired level of precision by utilizing a large number of parameters and FLOPs, which often leads to feature redundancy and makes it challenging to interpret the input images [22]. However, it is needless to expend computational resources on generating superfluous feature maps with a plethora of convolutional layers. Although lightweight network models, such as MobileNet [23], [24], [25], and SqueezeNet [26] reduce the number of FLOPs, the redundant feature maps generated by convolution are not effectively utilized. To construct networks that are suitable for low-computation platform operations, we have integrated GhostConv into the architecture of YOLOv5s. The GhostConv uses distributed extraction of feature maps to eliminate redundancy. If each original feature corresponds to S redundant features, the GhostConv only needs to generate N/S base features. It uses linear transformation to expand the original features and generate similar features. If the number of convolutional kernels is denoted by N, the size of the input feature map is $H \times W \times C$, the size of the output feature map is $H' \times W' \times N$, and the size of the convolutional kernel is $k \times k$, then the number of FLOPs required to perform conventional

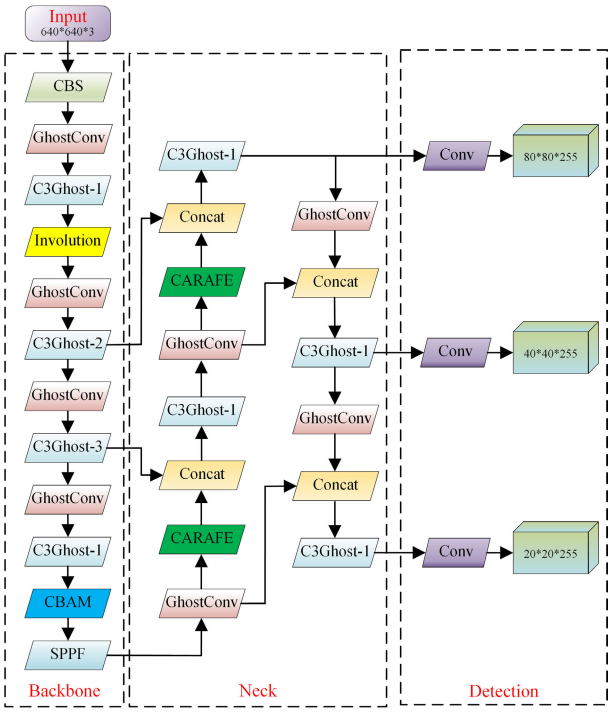

FIGURE 1. The improved YOLOv5s network model.

convolutional operations would be $N \times H \times W \times C \times k \times k$, where C denotes the number of input channels, H and W indicate the height and width of the input feature map, and H', W' represent the height and width of the output feature map, respectively. If the size of the convolution kernel for each linear operation is $d \times d$, the $Rate_S$ of the FLOPs for conventional convolution and GhostConv can be expressed as

$$\begin{aligned} &Rate_S \\ &= \frac{N \times H' \times W' \times C \times k \times k}{\frac{N}{S} \times H' \times W' \times C \times k \times k + (S-1) \times \frac{N}{S} \times H' \times W' \times d \times d} \\ &= \frac{C \times k \times k}{\frac{1}{S} \times C \times k \times k + \frac{S-1}{S} \times d \times d} \approx \frac{S \times C}{S + C - 1} \approx S \end{aligned} \quad (1)$$

The same parameter compression ratio $Rate_C$ is:

$$\begin{aligned} Rate_C &= \frac{N \times C \times k \times k}{\frac{N}{S} \times C \times k \times k + (S-1) \times \frac{N}{S} \times d \times d} \\ &\approx \frac{S \times C}{S + C - 1} \approx S \end{aligned} \quad (2)$$

As depicted in Figure 2, the Bottleneck structure in the original C3 is replaced by the GhostBottleneck, resulting in the formation of the new C3Ghost. The activation function of the Ghost module differs from that of ReLU function [27], as it uses the same SiLU function [28] as in Conv. The SiLU function has a smooth and non-monotonic lower term, without any upper term. This function is employed to prevent gradient vanishing as the number of network layers increases. Theoretical analysis has revealed that the number of FLOPs and parameters required for traditional convolutional extracted features is approximately S times that of the GhostConv. Therefore, the improved lightweight YOLOv5 model was constructed using GhostConv.

C. LIGHTWEIGHT BACKBONE NETWORK

The convolution operation shares convolution kernel parameters on the feature map of a single channel and uses different convolution kernels for different channels. This method is not conducive to extracting intricate features, and furthermore, convolution kernels exhibit redundancy in the channel dimension [29]. The involution operator differs from the convolution operator in that it shares parameters with the channel dimension and has different parameters in the spatial dimension, i.e., it is channel agnostic and spatially specific. In addition, the involution operator uses large convolution kernels to collect rich feature information in a large sensory field [30]. This feature improves the ability of the backbone network to discover different features at different locations and suppresses kernel redundancy better than ordinary convolution [31]. The involution operator significantly reduces the number of FLOPs and parameters of convolutional networks without changing the overall network architecture, providing a new strategy for network model optimization [32]. Thus, in the foundational model constructed by GhostConv, we endeavor to balance the complexity and performance of the lightweight network model by employing the Involution operator. Let $F \in R^{H \times W \times C_{in}}$ denote the input feature map, where H, W represent its height, width, and C_{in} is the number of input feature map channels. The generation of an involution kernel $I_{i,j}$ for each location of the input feature F_{in} is expressed as

$$C_m = C_{in}/r \quad (3)$$

$$I_{i,j} = \phi(f_{i,j}) = f_k\left(f_c(F_{i,j})\right) \quad (4)$$

$$f_c = SiLU(BN(Conv(\,))) \quad (5)$$

$$f_k = SiLU(BN(Conv(f_c))) \quad (6)$$

where C_{in} is the number of input feature map channels, C_m is the number of compressed channels, r is the channel reduction ratio. The function ϕ is the generating function of the involution kernel, consisting of the transformation matrix f_c and the transformation matrix f_k. Let $F_{i,j}$ be the feature tensor of size $1 \times 1 \times C_{in}$ located at coordinate (i, j) on the input feature map. Inside the cube of the feature tensor $F \in R^{H \times W \times C}$, each feature tensor $F_{i,j}$ located in the image lattice can be considered as a pixel representing certain high-level semantic patterns. After the feature map F_{in} is input to the involution operator, $F_{i,j}$ is extracted from F_{in} for channel compression, which is compressed from C_{in} to C_m to obtain the feature tensor $f_c \in \mathbb{R}^{1 \times 1 \times C_m}$. In this paper, the ReLU activation function of the original involution operator was replaced by using the SiLU activation function.

Figure 3 illustrates the calculation process of the involution operator. $F_{i,j}$ generates a feature tensor $F'_{i,j}$ of size $1 \times 1 \times k^2$ by means of the kernel-generating function ϕ (k is the size of the involution kernel), and expand it into the shape of the kernel by reshape operation to get the involution kernel on this

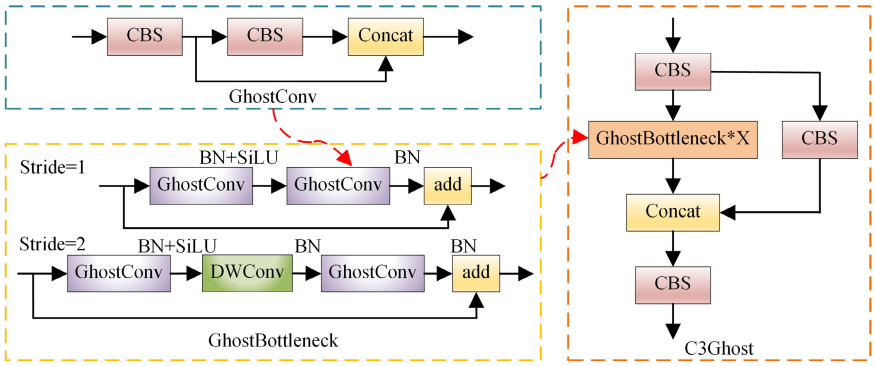

FIGURE 2. The structure of the C3Ghost module.

pixel point. Then the feature tensor of the neighborhood of the coordinate point (i, j) on the input feature map is multiplied and added with it to obtain the final output feature map F_{out}.

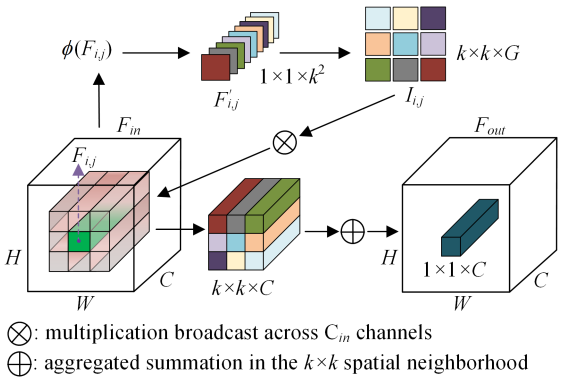

FIGURE 3. The calculation of the involution operator.

The involution operator possesses spatial specificity, which enables it to capture contextual information on a larger spatial scale compared with traditional convolution operators. It dynamically allocates network weights to prioritize the most significant visual information in the spatial domain of the image. Experimental evidence demonstrates that the addition of an involution operator reduces model complexity but results in a decline in detection precision due to channel modeling information is not sufficiently considered. Therefore, the constructed model must have a balance between precision and complexity.

D. ADDED ATTENTION MODULE

To enhance the feature extraction capability and compensate for the performance loss caused by the improvements, we incorporate an attention mechanism into the improved lightweight model. Attention mechanisms selectively highlight meaningful image information and suppress meaningless information [33]. Common attention mechanisms include squeeze-and-excitation (SE) attention [34], efficient channel attention (ECA) [35], and the CBAM [36]. The CBAM attention module is based on the SE attention mechanism and considers both spatial and channel dimensions,

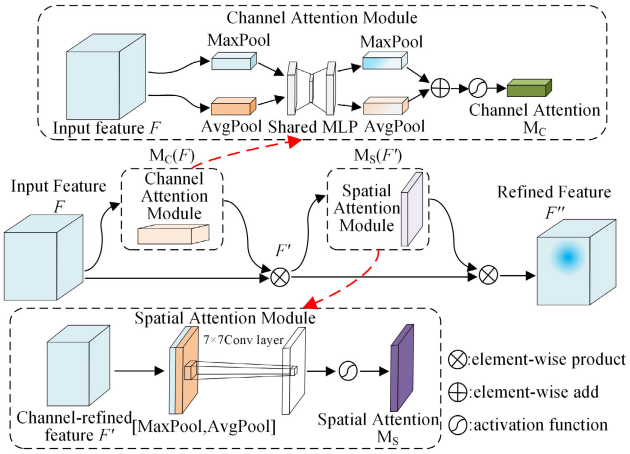

FIGURE 4. The structure of the CBAM.

thereby enhancing the network's ability to capture fine-grained features. Compared with the CBAM attention module, although the ECA module enhances the information exchange between channels, it does not improve the extraction capability of the network for fine-grained features. Therefore, the CBAM is utilized in the YOLOv5s network to obtain the key information for the current task to improve the efficiency and precision of image processing. It consists of the channel attention module (CAM) and the spatial attention module (SAM), whose structures are shown in Figure 4.

E. REPLACED UPSAMPLING METHOD

In this paper, the CARAFE operator was used to replace the upsampling method in the YOLOv5s network to design a lightweight and high-precision network for strawberry disease detection. The original upsampling module of the YOLOv5s uses a nearest-neighbor operator that generates noise in low-resolution images, resulting in potential feature loss. As shown in Figure 5, the CARAFE operator consists of the kernel prediction module (KPM) and the content-aware reassembly module (CaRM) [37]. It performs adaptive optimization of the reorganization kernel and feature reorganization for each pixel point to achieve content-aware upsampling. The CARAFE operator performs adaptive optimization of its recombination kernel based on the underlying content

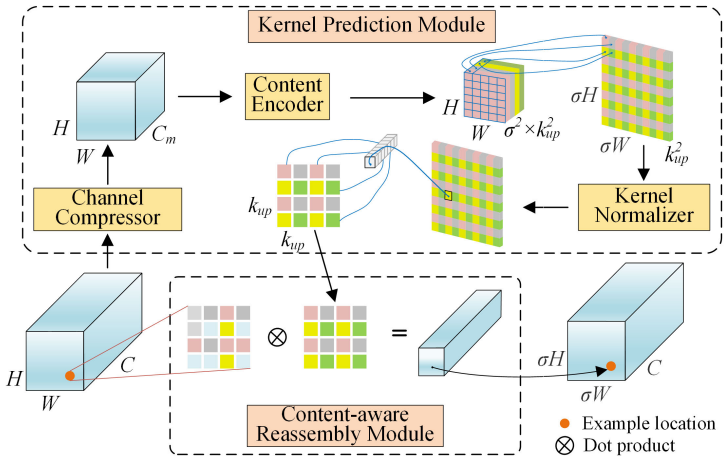

FIGURE 5. The structure of the CARAFE operator.

of the feature map. It then utilizes the recombination kernel to perform feature recombination at each pixel point, thereby achieving content-aware upsampling.

The CARAFE module takes a feature map of size $C \times H \times W$ as input to both its kernel prediction module and content-aware reassembly module, resulting in an output feature map of size $C \times \sigma H \times \sigma W$.The kernel prediction module consists of three submodules: the channel compressor program, the content encoder program and the kernel normalizer program. The channel compressor program uses 1×1 convolution to compress the number of channels of the input feature map to C_m. The content encoder program uses the $k_{encoder} \times k_{encoder} \times C_m \times C_{up}$ convolution to encode the compressed feature map to generate the reorganization kernel. Finally, the Softmax function of the kernel normalization program is used for normalization.

In the content-aware reassembly module, each feature point in the output feature map is mapped back to the input feature map. The $k_{up} \times k_{up}$ region centered on this point and the predicted upsampling kernel for it are then used in a dot product operations to compute the output value. Different channels at the same location share the same upsampling kernel. The CARAFE operator uses a weighted sum operation to reorganization local region features, assigning higher weights to features within the target region. This allows the sampling points to focused on the target region and ignore the background. Additionally, the upsampling kernel is generated in a content-aware manner, increasing the effective receptive field and allowing for the extraction of more information from the sampled point regions.

IV. EXPERIMENTS AND RESULTS

A. EXPERIMENTAL DETAILS AND EVALUATION METRICS

The experimental platform is based on NVIDIA RTX 2060 Super 8G GPU, Intel i5-12400F CPU@2.5GHz, and a Windows 10 operating system. The software includes CUDA11.5, cuDNN8.2, Python 3.9, and the Pytorch1.10.0 deep learning framework. The following hyperparameters were used: the momentum was 0.937, the initial learning rate was 0.01, the stochastic gradient descent (SGD) optimizer was used, the weight decay was 0.0005, and a warmup [38] procedure was used to mitigate model oscillation due to high initial learning rate during model training. The input images were enhanced online using hue-saturation-value (HSV) enhancement and mosaic enhancement [39] to increase the sample size. The input image size of the model was $640 \times 640 \times 3$, the batch size was 32, and the number of network training epochs was 300.

We used the number of parameters and FLOPs, model size, precision (P), recall (R), and mean Average Precision (mAP) to evaluate the algorithm's performance. The mAP is related to P and R and is defined as follows:

$$mAP = \frac{\sum_{i=1}^{N} \int_0^1 PdR}{N} \tag{7}$$

where the mAP is obtained by averaging the average precision of all categories, and N denotes the number of categories detected in the network. The precision is the ratio of the number of positive samples correctly predicted to the total number of positive samples:

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

The recall is the ratio of the number of positive samples correctly predicted to the number of all positive samples:

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

where true positives (TP), false positives (FP), and false negatives (FN) denote the number of correctly detected frames, falsely detected frames, and missed frames, respectively. The higher the value of mAP, the better the detection performance of the network model on the given dataset.

B. STRAWBERRY DISEASE DATASET

We evaluated the performance of the improved YOLOv5s model for strawberry disease detection in complex natural

FIGURE 6. Images of seven strawberry diseases.

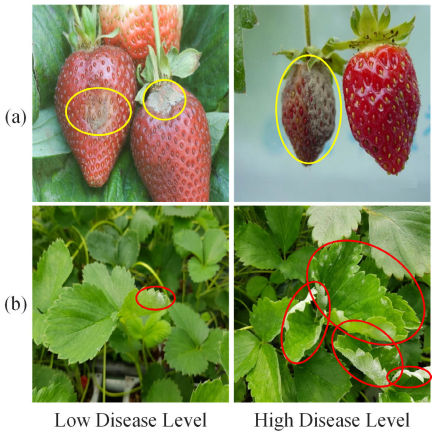

FIGURE 7. Images of different degrees of strawberry disease. (a) Gray Mold (b) Powdery Mildew Leaf (Mark the location of the disease with an ellipse circle).

environments using images from open-source datasets. The datasets used included the strawberry disease detection dataset shared by Afzaal et al. from Jeonbuk National University [40], and a dataset of strawberry leaf images for tipburn detection shared by Hariri and Avşar from Dukushele University [41]. Agricultural experts carefully selected and categorized the images, removing low-quality strawberry disease samples, in order to establish a high-quality strawberry disease dataset specifically designed for complex natural environments. The dataset was annotated using the LabelImg annotation tool. It consists of strawberry images affected by seven distinct disease categories, including the early, middle, and late stages of the diseases. Figure 6 provides examples of these seven categories of strawberry diseases.

The article provides examples of two common strawberry diseases, white powdery mildew and gray mold, shown in Figure 7. Strawberry white powdery mildew mainly affects leaves, petioles, flowers, pedicels, and fruits, and is characterized by the appearance of white powder on the infected area. Strawberry gray mold has the characteristics of fast transmission, rapid occurrence, and strong resistance to drugs. In the early stage of infection, the leaves are mostly diseased from the edge and spread inward. Infected strawberry flowers will quickly wither, and immature fruits will turn brown and become hardened. Infection of gray mold on fruits that are about to mature will cause the fruit to rot and generate a gray mold layer on the surface.

As shown in Table 1, the experimental data set consisted of 2246 images. The labeled data set was randomly divided into training, validation, and test sets using an 8:1:1 ratio. Various image augmentation techniques such as Mosaic, HSV enhancement, and others were used to increase the sample size and simulate natural environmental variations, including differences in illumination and target occlusion. This dataset was used to train the model and improve its ability to generalize to new, unseen data.

TABLE 1. Strawberry disease dataset.

Disease category	Number of images
Angular Leaf Spot	435
Blossom Blight	208
Calcium Deficiency of Leaves	378
Gray Mold	245
Leaf Spot	615
Powdery Mildew Fruit	163
Powdery Mildew Leaf	202

C. ADDITION OF INVOLUTION LAYERS

Compared with a traditional convolutional module, the GhostConv module exhibits thenumber of parameters and FLOPs that is only 1/S (where S is the compression ratio). We incorporated the GhostConv module into the YOLOv5s

backbone network and neck network (this model is called YOLO-G) to obtain a lightweight target detection network. On the base of the YOLO-G model, we take further action to lightweight the model. The involution operator was added to the backbone network of the YOLO-G model to investigate its effect on network performance. Comparison experiments were conducted to find the most suitable number and location of the involution modules. To be more specific, the involution layer was added to different positions in the network using specific combinations, which were: (1) added to the third, fifth, and seventh layers respectively, (2) added simultaneously to the third and fifth layers, and (3) added simultaneously to the third, fifth, and seventh layers. The number of FLOPs and mAP@0.5 were compared for the six models, and the experimental results are presented in Table 2 (Experiment 1 using the YOLO-G model). Experiments 1, 2, 5, and 6 demonstrate that incorporate multiple involution layers significantly reduce the number of FLOPs and the mAP@0.5 value. However, the convergence speed of the model is slower. Experiment 6, which utilizes three involution layers, has the largest impact on the model performances, and the number of FLOPs is reduced by 71.1% and mAP@0.5 by 4.9%, compared with the Experiment 1. Experiments 2, 3, and 4 indicate that the closer the involution layer is to the end of the network, the higher the number of FLOPs and the lower the mAP@0.5. Adding one involution layer (Experiment 2) achieves a better balance between the number of FLOPs and detection precision, and the number of FLOPs and mAP@0.5 are reduced by 55.4% and 1%, respectively, compared with the YOLO-G model. The involution operator and convolution operator were combined to obtain a good balance between model complexity and detection performance. Therefore, Experiment 2 provided the optimum results; it is referred to as YOLO-GI.

TABLE 2. Results for adding the involution layer at different locations.

No.	Third	Fifth	Seventh	FLOPs (G)	mAP@0.5 (%)
1				8.3	**94.0**
2	√			**3.7**	93.0
3		√		4.2	90.8
4			√	8.3	90.5
5	√	√		2.4	89.9
6	√	√	√	2.4	89.1

D. ADDITION OF ATTENTION MODULES

The CARAFE operator is a type of upsampling operator that is designed to capture environmental feature information from large receptive fields and has been shown to improve the performance of lightweight models. We leveraged the CARAFE operator's capacity to obtain environmental feature information from large receptive fields to offset the potential performance degradation resulting from model lightweighting.

Specifically, we replaced the upsampling operator of the YOLO-GI model with the CARAFE operator, and named the resulting architecture as YOLO-GIC. The corresponding results are presented in Table 3. The precision increased by 1.4%, the recall increased by 2.4%, but the mAP@0.5 decreased by 0.3% after replacing the nearest-neighbor upsampling module with the CARAFE operator. This can be attributed to the fact that the lightweight improvement strategy reduced the feature extraction capability of the backbone network, and the CARAFE operator adjusted the upsampling kernel based on the input feature map, the fine-grained feature information was not adequately considered, decreasing the mAP@0.5 value.

TABLE 3. Results for adding carafe modules.

Scheme	CARAFE	Precision (%)	Recall (%)	mAP@0.5 (%)
YOLO-GI		92.4	86.8	93.0
YOLO-GIC	√	93.8	89.2	92.7

We investigated the effect of adding the attention modules, to counteract performance degradation resulting from lightweight improvements to the model. Specifically, we added the ECA, coordinate attention (CA) [42], and shuffle attention (SA) [43] modules, and the CBAM to the YOLO-GIC model, and conducted experiments to evaluate their efficacy. The results, as presented in Table 4, indicate that the CBAM module (Experiment 4) achieved the best performance in terms of precision, recall, and mAP@0.5. The resulting architecture, which was called YOLO-GIC-C, showed a 0.9% increase in precision, 3.5% increase in recall, and 1.7% increase in mAP@0.5 compared with the YOLO-GI model, thus indicating that the improvement strategy did not lead to a performance loss.

TABLE 4. Results for adding attention modules.

No.	Attention model	Precision (%)	Recall (%)	mAP@0.5 (%)
1	ECA	91.1	89.6	92.7
2	CA	91.1	89.6	92.7
3	SA	93.5	85.7	92.5
4	CBAM	93.3	**90.3**	**94.7**

E. ABLATION EXPERIMENT

Ablation experiments were conducted on the proposed model to evaluate the contributions of the improvements on the performance of the strawberry disease detection model. We added the GhostConv module, involution layer, CARAFE operator, and CBAM to the baseline model YOLOv5s, and the other training parameters were consistent with the final proposed model. The results of the ablation experiments are listed in Table 5, where " √ " indicates that the improvement strategy is included in the network.

The results show that the addition of the GhostConv module to the YOLOv5s network reduced the number of parameters and FLOPs by 47.9% and 48.1%, respectively. The incorporation of the involution layer alone reduced the number of FLOPs and recall by 62.5% and 2.2% and increased

TABLE 5. Results of ablation experiments.

Scheme	GhostConv	Involution	CARAFE	CBAM	Parameters (M)	FLOPs (G)	Size (M)	Precision (%)	Recall (%)	mAP@0.5 (%)	FPS
A					7.1	16.0	14.1	88.8	91.3	93.3	93.5
B	√				3.7	8.3	7.8	89.1	91.3	94.0	117.6
C		√			7.1	6.0	14.3	93.6	89.1	93.5	129.9
D	√	√			3.3	3.7	7.7	92.4	86.8	93.0	112.3
E	√		√		3.9	8.6	8.2	92.7	90.2	93.7	113.6
F		√	√		7.2	6.0	14.6	92.3	90.1	93.3	114.9
G	√	√	√		3.9	3.5	8.0	93.8	89.2	92.7	121.9
H	√	√		√	3.8	3.3	7.8	90.5	89.0	92.8	116.2
I	√		√	√	3.9	8.7	8.2	92.2	89.9	93.7	102.1
J	√	√	√	√	3.9	3.6	8.1	93.3	90.3	94.7	92.6

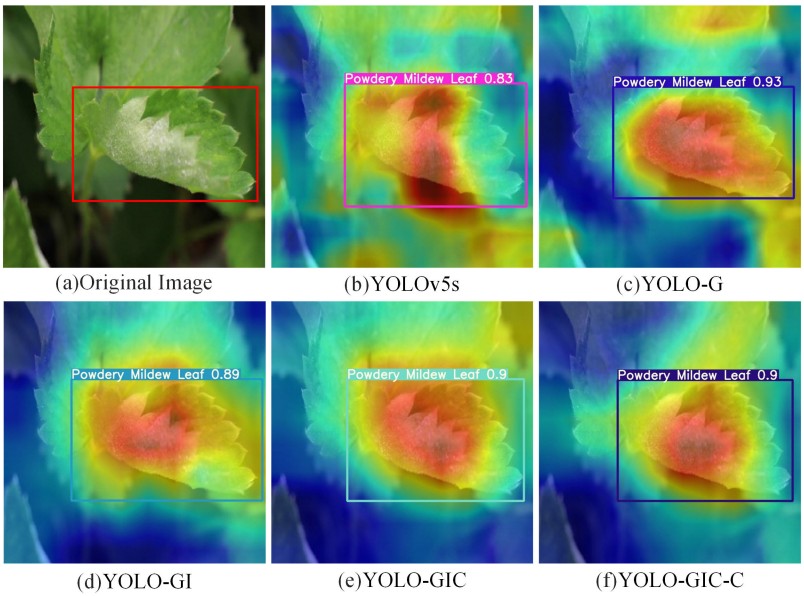

FIGURE 8. Regions of interest obtained from different algorithms.

the precision by 4.8%. The addition of the GhostConv module and involution layer reduced the number of parameters and FLOPs, the recall, and mAP@0.5 by 53.5%, 76.9%, 4.5%, and 0.3%, respectively, and increased the precision by 3.6%. These results indicate that the addition of the GhostConv module and involution operator significantly reduced the number of parameters and FLOPs, making it feasible for deployment on resource-constrained mobile devices but also reducing the detection performance of the network. The final model that included the CARAFE module and CBAM reduced the number of parameters and FLOPs, the weight size of the network, and the recall by 45.0%, 77.5%, 42.6%, and 1.0%, respectively. However, it increased the precision and mAP@0.5 by 4.5% and 1.4%, respectively, compared with the baseline model (YOLOv5s). On the strawberry disease dataset, the lightweight model proposed in this study achieved a precision of 93.3%, a recall of 90.3%, a mAP@0.5 of 94.7%, and an FPS of 92.6, with 3.9M parameters and 3.6 G FLOPs. This result shows that the proposed YOLO-GIC-C model achieved the best balance between model performance and model complexity with a minor performance decrease.

The last layer of the feature map of the deep CNN contains numerous high-level semantic features and detailed spatial information. Thus, a visualization of the last layer of the feature map illustrates which part of an image contributes more to the final output of the model. To achieve this, we used gradient-weighted class activation mapping (Grad-CAM) [44] to visualize the class activation map. In Figure 8, the area highlighted in red represents the region of interest where the network is concentrating its attention. It serves as the primary basis for evaluating the model's performance in terms of its ability to detection disease features. As depicted in Figure 8 (b) and (c), the YOLO-G network effectively covers the region of interest, demonstrating the feasibility of adding the GhostConv module to the YOLOv5s network. As shown in Figure 8 (c) and (d), it is evident that the class activation maps generated by the YOLO-GI network have a more concentrated region of interest, but a smaller coverage area of disease regions. This suggests that the incorporation of the Involution operator enables the backbone network to aggregate contextual information over a larger spatial area. However, channel dimension sharing reduces

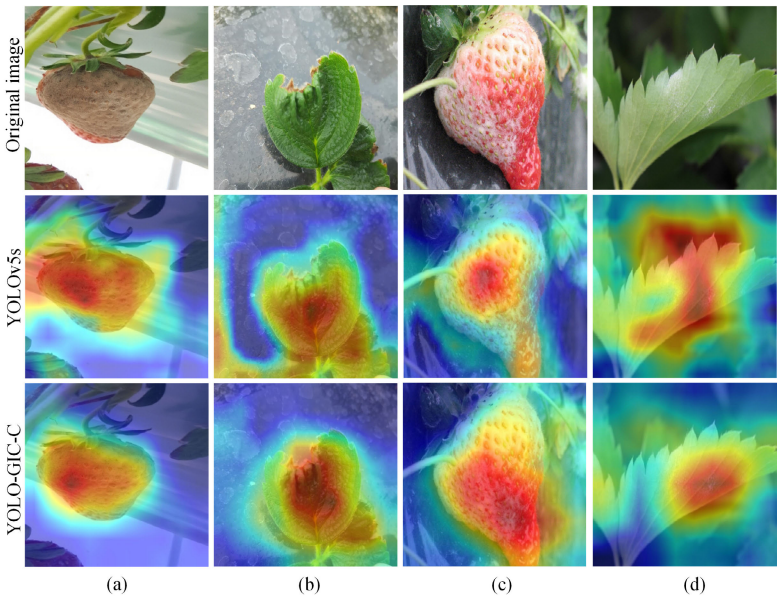

FIGURE 9. Comparison of class activation maps for different strawberry diseases derived from different models. (a) Gray mold; (b) Calcium deficiency of leaves; (c) Powdery mildew fruit; (d) Powdery mildew leaf.

TABLE 6. Performance results for different lightweight networks.

Model	Parameters (M)	FLOPs (G)	Size (M)	Precision (%)	Recall (%)	mAP@0.5 (%)
YOLOv3-Tiny[46]	8.7	12.9	17.5	75.5	81.6	81.8
YOLOv4-Mish	8.2	20.6	18.7	91.7	89.7	94.1
PPYOLOE-S[47]	8.4	13.9	17.2	92.4	86.8	93.5
YOLOv5n	**1.8**	4.2	**3.9**	89.1	83.6	90.8
YOLOv5Lite-C[48]	4.4	8.6	9.2	92.6	85.4	90.4
YOLOX-Tiny[49]	4.6	12.3	9.3	92.4	88.5	93.9
YOLOX-S	8.1	16.3	16.3	93.2	88.5	93.7
YOLOv7-Tiny[50]	6.1	13.2	12.3	85.8	85.2	89.4
YOLOv8n[51]	3.1	8.2	6.3	93.1	85.8	93.5
YOLOv8s	11.2	28.8	22.5	94.9	86.2	94.3
YOLO-GIC-C(Ours)	3.9	**3.6**	8.1	93.3	**90.3**	**94.7**

the feature extraction capability of the network. As shown in Figure 8 (d) and (e), the region of interest of the YOLO-GIC network better represents the characteristic regions of the powdery mildew on the leaf, indicating that the CARAFE operator that performs content-based upsampling improves the network's ability to capture detailed features. As shown in Figure 8(e) and (f), the region of interest of the YOLO-GIC-C network better delineates the disease region, illustrating that the CBAM improved the network's feature extraction ability and suppressed non-critical information. A comparison of Figure 8(b) and (f) shows that the YOLOv5s mistakenly assumed that powdery mildew occurred at the leaf edges, whereas the proposed YOLO-GIC-C model focused on the main characteristics of the disease and accurately delineated the region of interest containing disease characteristics.

As shown in Figure 9, the regions of interest in the class activation maps generated by the YOLOv5s network did not focus on the features related to strawberry disease, while the YOLO-GIC-C model accurately highlighted the affected regions. The proposed strawberry disease detection model demonstrated better proficiency in capturing the strawberry disease features compared to the YOLOv5s network, and effectively avoided the loss of target features, showcasing its superior performance for strawberry disease detection.

F. COMPARISON WITH RELATED METHODS

The performance of the proposed model was compared with that of other algorithms such as YOLOv3-tiny, YOLOv4-Mish, and PPYOLOE-S, and the results are listed in Table 6. The YOLO-GIC-C model has the second-smallest number of parameters and weight size after YOLOv5n and YOLOv8n, the lowest number of FLOPs, and the highest mAP@0.5 and recall. These results demonstrate that the proposed lightweight target detection algorithm is superior to comparable algorithms, exhibiting higher precision, fewer FLOPs, and lightweight architecture.

The proposed lightweight model was also evaluated on the PlantDoc dataset [45], which is a publicly available

TABLE 7. Performance results of different lightweight networks on PlantDoc dataset.

Model	Parameters (M)	FLOPs (G)	Size (M)	Precision (%)	Recall (%)	mAP@0.5 (%)
YOLOv5n	1.8	4.2	**3.9**	43.6	41.5	26.2
YOLOv5s	7.1	16.0	14.1	42.9	42.1	27.0
YOLOv7-Tiny	6.1	13.2	12.3	40.1	40.4	25.4
YOLOv8n	3.1	8.2	6.3	43.4	41.7	27.8
YOLOv8S	11.2	28.8	22.5	41.3	44.5	30.2
YOLO-GIC-C(Ours)	3.9	**3.6**	8.1	**44.5**	42.4	27.9

dataset containing images of 13 plant species and 27 categories (17 categories of diseased leaves and 10 categories of healthy leaves) The results in Table 7 show that the proposed lightweight model outperforms YOLOv5s and achieves a precision, recall, and mAP@0.5 of 44.5%, 42.4%, and 27.9%, respectively. Additionally, compared with other models, the proposed model achieved the lowest FLOPs, highest precision, second-highest recall, and second-highest mAP@0.5. Although the precision and mAP@0.5 of the proposed model were slightly lower than YOLOv8s, its number of parameters, FLOPs, and model size were only 34.8%, 12.5%, and 36% of YOLOv8s, respectively. The comparison between the two datasets also revealed that the PlantDoc dataset has more detection categories but fewer images per category. Overall, the proposed model structure can be retrained for different datasets or scenarios to meet the real-time detection needs of plant diseases. By comparing the PlantDoc dataset and the strawberry disease dataset, it was observed that the former has more detection categories but fewer images per category. Furthermore, YOLOv8s model has a substantially higher number of parameters, FLOPs, and weight size than the proposed lightweight model. In practical application scenarios, the proposed model structure can be retrained for single or multiple different datasets or scenarios to meet the real-time detection requirements of plant diseases.

V. DISCUSSION

The strawberry disease dataset contains images of 7 different types of strawberry diseases at various stages of infection in complex natural environments. According to Table 8, the lightweight model proposed in this paper demonstrates effective performance in detecting angular leaf spot, blossom blight, and calcium deficiency of leaves. However, the recall rates for powdery mildew fruit and powdery mildew leaf are relatively low at 80.3% and 80.7%, respectively, which are lower than the average recall rate by 10% and 9.6%, respectively. Through the analysis of the dataset and the observation of the results on the test set, it was found that the severity of strawberry disease infection and the environment have a significant impact on detection performance. Due to the different background environments in each category and the indistinct disease characteristics that are easily affected by lighting, the number of false positives and false negatives has increased to a large extent.

For example, as illustrated in Figure 10 (a), gray mold was predicted as powdery mildew fruit due to lighting effects. The similar white disease characteristics of gray mold and powdery mildew fruit resulted in the misclassification of gray mold samples as powdery mildew fruit samples. Figure 10 (b) and (c) show the reasons for the undetected white powdery mildew disease. The early features of fruit powdery mildew and leaf powdery mildew are not obvious, and the low distinguishability of leaf powdery mildew features and background environments under the light has led to model omissions.

TABLE 8. Performance results for detecting different strawberry diseases.

Disease category	Precision(%)	Recall(%)	mAP@0.5(%)
Angular Leaf Spot	96.6	94.0	98.1
Blossom Blight	97.6	98.1	99.5
Calcium Deficiency of Leaves	94.4	95.5	97.8
Gray Mold	94.7	92.3	93.7
Leaf Spot	90.4	91.6	96.5
Powdery Mildew Fruit	88.3	80.3	87.4
Powdery Mildew Leaf	91.1	80.7	89.7
All	93.3	90.3	94.7

FIGURE 10. False positive and false negative display for strawberry disease detection. (a) Gray mold, (b) Powdery mildew fruit, (c) Powdery mildew leaf. (false positive and false negative targets are marked with red and yellow ovals, respectively).

The proposed lightweight model for the real-time detection of strawberry diseases in greenhouses, even in complex environments, shows excellent potential. It has achieved 93.3% precision and 94.7% mAP@0.5, which were 4.5% and 1.4% higher, respectively, than those of the YOLOv5s network. This model exhibited significantly lower computational complexity than comparable models, whereas providing high precision, enabling farmers to detect strawberry diseases timely and accurately to prevent disease spread. A camera is indispensable for acquiring the input images. Thus, the image quality affects the modeling accuracy. Early signs of strawberry diseases may be difficult to detect, and some images had blurred backgrounds and overexposure,

resulting in false or missed detections. Therefore, the quality of images requires improvement, and richer datasets are necessary for enhancing detection performance. Furthermore, different camera deployment schemes are needed for different strawberry diseases, minimizing the risk of interference by non-strawberry disease targets.

VI. CONCLUSION

The paper proposed a lightweight strawberry disease detection network called YOLO-GIC-C, an improvement of YOLOv5s, for plant disease management. The model enables the real-time detection of strawberry diseases in greenhouses using mobile devices at low cost. The YOLOv5s was improved to maintain a balance between model complexity and model performance. The GhostConv module and involution operator expanded the effective receptive field of the CNN, reducing the number of parameters and FLOPs and the weight size of the network by 53.5%, 76.9%, and 45.4%, respectively, but decreasing model performance. The CBAM and CARAFE module enhanced the model detection performance by focusing on meaningful information on strawberry diseases and suppressing unimportant information. On the strawberry disease dataset, the final model had 45.0% fewer parameters, 77.5% fewer FLOPs, and 42.6% lower weight size than the YOLOv5s model, and the precision and mAP@0.5 were 4.5% and 1.4% higher, respectively. The mAP@0.5 of the improved model was 12.9%, 0.6%, 1.2%, 3.9%, 4.3%, 0.8%, 1%, 5.3%, 1.2%, and 0.4%, higher, and the number of FLOPs were 72.1%, 82.5%, 74.1%, 14.3%, 58.1%, 70.7%, 77.9%, 72.7%, 56.1%, and 87.5% lower than that of mainstream target detection algorithms (YOLOv3-Tiny, YOLOv4-Mish, PPYOLOE-S, YOLOv5n, YOLOv5Lite-C, YOLOX-Tiny, YOLOX-S, YOLOv7-Tiny, YOLOv8n, YOLOv8s, respectively). The results demonstrate that the proposed algorithm has significantly lower hardware requirements than comparable models, demonstrating its excellent applicability for plant disease identification in complex environments. In future research, we plan to improve the detection performance by optimizing the network structure and adopting the latest high-performance benchmark model. Furthermore, we plan to deploy the model in embedded mobile devices for the real-time detection of strawberry diseases in greenhouses, making it more accessible and convenient for farmers.

REFERENCES

[1] S. Petrasch, S. J. Knapp, J. A. L. van Kan, and B. Blanco-Ulate, "Grey mould of strawberry, a devastating disease caused by the ubiquitous necrotrophic fungal pathogenBotrytis cinerea," *Mol. Plant Pathol.*, vol. 20, no. 6, pp. 877–892, Apr. 2019.

[2] X. Wang, X. Zhang, and G. Zhou, "Automatic detection of rice disease using near infrared spectra technologies," *J. Indian Soc. Remote Sens.*, vol. 45, no. 5, pp. 785–794, Oct. 2017.

[3] J. A. Wani, S. Sharma, M. Muzamil, S. Ahmed, S. Sharma, and S. Singh, "Machine learning and deep learning based computational techniques in automatic agricultural diseases detection: Methodologies, applications, and challenges," *Arch. Comput. Methods Eng.*, vol. 29, no. 1, pp. 641–677, Jan. 2022.

[4] J. Chen, D. Zhang, A. Zeb, and Y. A. Nanehkaran, "Identification of rice plant diseases using lightweight attention networks," *Exp. Syst. Appl.*, vol. 169, May 2021, Art. no. 114514.

[5] H. K. Suh, J. W. Hofstee, and E. J. van Henten, "Investigation on combinations of colour indices and threshold techniques in vegetation segmentation for volunteer potato control in sugar beet," *Comput. Electron. Agricult.*, vol. 179, Dec. 2020, Art. no. 105819.

[6] J. Shin, Y. K. Chang, B. Heung, T. Nguyen-Quang, G. W. Price, and A. Al-Mallahi, "A deep learning approach for RGB image-based powdery mildew disease detection on strawberry leaves," *Comput. Electron. Agricult.*, vol. 183, Apr. 2021, Art. no. 106042.

[7] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, "A survey of deep learning and its applications: A new paradigm to machine learning," *Arch. Comput. Methods Eng.*, vol. 27, no. 4, pp. 1071–1092, Sep. 2020.

[8] S. R. Dubey and A. S. Jalal, "Adapted approach for fruit disease identification using images," *Int. J. Comput. Vis. Image Process.*, vol. 2, no. 3, pp. 44–58, Jul. 2012.

[9] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Comput. Electron. Agricult.*, vol. 145, pp. 311–318, Feb. 2018.

[10] T. Anandhakrishnan and S. M. Jaisakthi, "Deep convolutional neural networks for image based tomato leaf disease detection," *Sustain. Chem. Pharmacy*, vol. 30, Dec. 2022, Art. no. 100793.

[11] X. Nie, L. Wang, H. Ding, and M. Xu, "Strawberry verticillium wilt detection network based on multi-task learning and attention," *IEEE Access*, vol. 7, pp. 170003–170011, 2019.

[12] J.-R. Xiao, P.-C. Chung, H.-Y. Wu, Q.-H. Phan, J.-L.-A. Yeh, and M. T.-K. Hou, "Detection of strawberry diseases using a convolutional neural network," *Plants*, vol. 10, no. 1, p. 31, Dec. 2020.

[13] Y. Li, J. Wang, H. Wu, Y. Yu, H. Sun, and H. Zhang, "Detection of powdery mildew on strawberry leaves based on DAC-YOLOv4 model," *Comput. Electron. Agricult.*, vol. 202, Nov. 2022, Art. no. 107418.

[14] I. H. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," *Social Netw. Comput. Sci.*, vol. 2, no. 6, p. 420, Aug. 2021.

[15] S. Xiang, Q. Liang, W. Sun, D. Zhang, and Y. Wang, "L-CSMS: Novel lightweight network for plant disease severity recognition," *J. Plant Diseases Protection*, vol. 128, no. 2, pp. 557–569, Apr. 2021.

[16] J. Chen, W. Chen, A. Zeb, S. Yang, and D. Zhang, "Lightweight inception networks for the recognition and detection of rice plant diseases," *IEEE Sensors J.*, vol. 22, no. 14, pp. 14628–14638, Jul. 2022.

[17] F. Qi, Y. Wang, and Z. Tang, "Lightweight plant disease classification combining GrabCut algorithm, new coordinate attention, and channel pruning," *Neural Process. Lett.*, vol. 54, no. 6, pp. 5317–5331, Dec. 2022.

[18] (2021). *Ultralytics/YOLOv5: V6.0*. [Online]. Available: https://zenodo.org/record/5563715#.ZEJHFeZBw6Q

[19] H. Park, Y. Yoo, G. Seo, D. Han, S. Yun, and N. Kwak, "C3: Concentrated-comprehensive convolution and its application to semantic segmentation," 2018, *arXiv:1812.04920*.

[20] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.

[21] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.

[22] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1577–1586.

[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[24] A. Howard, M. Sandler, B. Chen, W. Wang, L. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[25] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[26] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size," 2016, *arXiv:1602.07360*.

[27] G. Xavier, B. Antoine, and B. Yoshua, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist. (AISTATS)*, vol. 15, Jun. 2011, pp. 315–323.
[28] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Netw.*, vol. 107, pp. 3–11, Nov. 2018.
[29] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 1–11.
[30] X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31 × 31: Revisiting large kernel design in CNNs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 11963–11975.
[31] X. Wang and S. X. Yu, "Tied block convolution: Leaner and better CNNs with shared thinner filters," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, May 2021, pp. 10227–10235.
[32] D. Li, J. Hu, C. Wang, X. Li, Q. She, L. Zhu, T. Zhang, and Q. Chen, "Involution: Inverting the inherence of convolution for visual recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12316–12325.
[33] G. Liu, Y. Hu, Z. Chen, J. Guo, and P. Ni, "Lightweight object detection algorithm for robots with improved YOLOv5," *Eng. Appl. Artif. Intell.*, vol. 123, Aug. 2023, Art. no. 106217.
[34] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 2011–2023, Aug. 2020.
[35] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-Net: Efficient channel attention for deep convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11531–11539.
[36] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 3–19.
[37] J. Wang, K. Chen, R. Xu, Z. Liu, C. C. Loy, and D. Lin, "CARAFE: Content-aware reassembly of features," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3007–3016.
[38] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu, "On layer normalization in the transformer architecture," in *Proc. 37th PMLR*, 2020, pp. 10524–10533.
[39] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.
[40] U. Afzaal, B. Bhattarai, Y. R. Pandeya, and J. Lee, "An instance segmentation model for strawberry diseases based on mask R-CNN," *Sensors*, vol. 21, no. 19, p. 6565, Sep. 2021.
[41] M. Hariri and E. Avçar, "Tipburn disorder detection in strawberry leaves using convolutional neural networks and particle swarm optimization," *Multimedia Tools Appl.*, vol. 81, no. 8, pp. 11795–11822, Mar. 2022.
[42] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13708–13717.
[43] Q. Zhang and Y. Yang, "SA-Net: Shuffle attention for deep convolutional neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 2235–2239.
[44] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, Feb. 2020.
[45] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, "PlantDoc: A dataset for visual plant disease detection," in *Proc. 7th ACM IKDD CoDS 25th COMAD*, Jan. 2020, pp. 249–253.
[46] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.
[47] S. Xu, X. Wang, W. Lv, Q. Chang, C. Cui, K. Deng, G. Wang, Q. Dang, S. Wei, Y. Du, and B. Lai, "PP-YOLOE: An evolved version of YOLO," 2022, *arXiv:2203.16250*.
[48] *YOLOv 5-Lite: Lighter, Faster, and Easier to Deploy*. Accessed: 2021. [Online]. Available: https://github.com/ppogg/YOLOv5-Lite
[49] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO series in 2021," 2021, *arXiv:2107.08430*.
[50] C.-Y. Wang, A. Bochkovskiy, and H.-Y. Mark Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," 2022, *arXiv:2207.02696*.
[51] (2023). *YOLO by Ultralytics*. [Online]. Available: https://github.com/ultralytics/ultralytics

SHUNLONG CHEN was born in Mianyang, Sichuan, China, in 1998. He is currently pursuing the M.S. degree in mechanical engineering with the Sichuan University of Science and Engineering, Zigong, China.

His current research interests include deep learning and object detection and their applications in plant disease detection.

YINGHUA LIAO was born in Yibin, Sichuan, China, in 1976. He received the B.S. degree in process and equipment of machinery manufacturing and the M.S. degrees in mechanical design, manufacturing, and automation from Sichuan University, Chengdu, China, in 1999 and 2005, respectively, and the Ph.D. degree in mechanical engineering from Chongqing University, Chongqing, China, in 2019.

Since 1999, he has been focusing on intelligent equipment design and its application. He has authored or coauthored more than 30 articles and holds 17 patents.

FENG LIN was born in Putian, Fujian, China, in 1998. He is currently pursuing the M.S. degree in mechanical engineering with the Sichuan University of Science and Engineering, Zigong, China.

His current research interests include industrial image processing, computer vision, and machine learning.

BO HUANG was born in Neijiang, Sichuan, China, in 1978. He received the B.S. degree in mechanical engineering and automation from the Taiyuan University of Technology, Taiyuan, China, in 2002, and the M.S. degree in machine design theory from the Sichuan University of Science and Engineering, Zigong, China, in 2013.

Since 2002, he has been an Assistant Professor and then an Associate Professor with the Sichuan University of Science and Engineering. His research interests include mechanical design and manufacture, mechatronics, and embedded systems.