*machines*

*Article*

# A Neural Network Structure with Attention Mechanism and Additional Feature Fusion Layer for Tomato Flowering Phase Detection in Pollination Robots

**Tongyu Xu [1], Xiangyu Qi [1], Sen Lin [2,*], Yunhe Zhang [2], Yuhao Ge [2], Zuolin Li [2], Jing Dong [3] and Xin Yang [1]**

[1] College of Information and Electrical Engineering, Shenyang Agricultural University, Shenyang 110866, China
[2] Research Center of Intelligent Equipment, Beijing Academy of Agriculture and Forestry Sciences, Beijing 100097, China
[3] Nongxin Science & Technology (Beijing) Co., Ltd., Beijing 100097, China
* Correspondence: linseng@nercita.org.cn

**Abstract:** In recent years, convolutional neural networks have made many advances in the field of computer vision. In smart greenhouses, using robots based on computer vision technology to pollinate flowers is one of the main methods of pollination. However, due to the complex lighting environment and the influence of leaf shadow in the greenhouse, it is difficult for the existing object detection algorithms to have high recall rate and accuracy. Based on this problem, from the perspective of application, we proposed a Yolov5s-based tomato flowering stage detection method named FlowerYolov5, which can well identify the bud phase, blooming phase and first fruit phase of tomato flowers. Firstly, in order to reduce the loss of tomato flower feature information in convolution and to strengthen the feature extraction of the target, FlowerYolov5 adds a new feature fusion layer. Then, in order to highlight the information of the object, the Convolutional Block Attention module (CBAM) is added to the backbone layer of FlowerYolov5. In the constructed tomato flower dataset, compared with YOLOv5s, the mAP of FlowerYolov5 increased by 7.8% (94.2%), and the $F_1$ score of FlowerYolov5 increased by 6.6% (89.9%). It was found that the overall parameter of FlowerYolov5 was 23.9 Mbyte, thus achieving a good balance between model parameter size and recognition accuracy. The experimental results show that the FlowerYolov5 has good robustness and more accurate precision. At the same time, the recall rate has also been greatly improved. The prediction results of the proposed algorithm can provide more accurate flower positioning for the pollination robot and improve its economic benefits.

**Keywords:** robot technology; pollination robot; tomato flowering phase detection; attention mechanism; deep learning method

check for updates

**Citation:** Xu, T.; Qi, X.; Lin, S.; Zhang, Y.; Ge, Y.; Li, Z.; Dong, J.; Yang, X. A Neural Network Structure with Attention Mechanism and Additional Feature Fusion Layer for Tomato Flowering Phase Detection in Pollination Robots. *Machines* **2022**, *10*, 1076. https://doi.org/10.3390/machines10111076

Academic Editor: Antonio J. Marques Cardoso

Received: 18 October 2022
Accepted: 12 November 2022
Published: 15 November 2022

## 1. Introduction

Tomatoes are the fourth largest vegetable crop planted in China, with an annual output of 55 million tons, accounting for 7% of the total vegetable output. With the continuous development of agricultural information technology, the methods for growing tomatoes are gradually shifting from traditional to intelligent and precision agriculture [1,2]. Intelligent and precision agriculture combine modern science and technology to achieve the automated and intelligent planting and management of tomatoes, thereby improving their yield and quality. With the support of IoT technology [3], artificial intelligence technology [4,5], cloud computing, and big data computing, intelligent greenhouses have become one of the main ways to grow tomatoes. The use of robots for pollination operations is an essential part of achieving intelligence in greenhouses, which can save labor costs and improve pollination efficiency. Therefore, a detection model that can accurately identify the flowering phase and the area where the flowers are located is needed to improve the yield and quality of tomatoes in intelligent greenhouses.

Traditional flower detection methods mainly use algorithms such as image filtering, feature fusion, and edge detection to extract the feature information of flowers. Indeed, scholars have conducted extensive research on such methods. For example, Ashraf Ahmad et al. [6] achieved the detection of flower regions from images and identified flower species by combining the color, texture, and shape features of different flower images. Aleya et al. [7] used the *k*-means algorithm to separate flowers from the background and completed the detection of broken flowers based on the histogram distribution. Dorj et al. [8] used a Gaussian filter to reduce the effects of noise and illumination in order to achieve the accurate detection of citrus flowers. Most of these methods for detecting flowers require a single, simple background, making the flower objects easily detectable from the background using features such as color and texture. However, the complex environment in an intelligent greenhouse, often with serious occlusion problems, increases the difficulty of detecting flowers.

Object detection technology based on deep learning [9,10] has been applied to flower detection with good results, due to its good generalization ability, high detection accuracy, and fast speed. The common deep learning object detection research can be divided into one-stage and two-stage networks. The first-level network of a two-stage object detector is used for generating some candidate boxes, while the second-level network classifies each candidate box and corrects its position. Chen et al. [11] chose a faster region-based convolutional neural network (R-CNN) to achieve detection and counting of strawberry flowers and ripe and unripe strawberries. An improved convolutional neural network (CNN)-based method for tomato flower and fruit detection was proposed by Sun et al. [12]. Compared with the original Faster R-CNN algorithm, the detection accuracy was significantly improved by using Resnet-50 with residual blocks instead of the conventional vgg16 feature extraction network. Sun et al. [13] have improved DeepLab-ResNet, a semantic segmentation-based network, to detect apple, peach, and pear flowers. Saad et al. [14] have proposed a Faster R-CNN-based model for detecting pepper fruit and flowers, which optimizes the parameters involved in classifying and detecting peppers and flowers. Chu et al. [15] have proposed a two-stage detection network to achieve the real-time detection and capturing of objects.

Compared to two-stage networks [16,17], a one-stage network [18,19] has a faster detection speed, shorter training time, and can reduce the negative samples generated by complex backgrounds, making them more suitable for use in intelligent greenhouses. Huang et al. [20] used CSPDarknet53 as the backbone of the original Yolov3, and used CIOU as the regression mechanism of their model to detect immature apples. Tian et al. [21] have proposed a flower detection network based on an SSD algorithm using a gradient descent algorithm with the Adam optimization function, improving the model convergence rate and increasing the accuracy. Cheng et al. [22] have proposed an end-to-end flower detection method based on the Yolov4 object detection model. The model's operations on invalid features were reduced by using an attention mechanism and optimizing the loss function. The method was tested using the Oxford University flower data set, and 84% and 94% confidence were reached in sunflower and cherry blossom detection, respectively.

In summary, most of the current object detection models for pollination robots in intelligent greenhouses can only identify flowers, while detection of the flowering phase is still a less-researched direction. Moreover, application of the above-mentioned detection models to pollination robots makes them susceptible to occlusion and complex backgrounds, resulting in the model detecting the flowering phase with low accuracy and high misdetection rates. This results in some flowers not being pollinated properly, leading to economic losses.

In recent years, attention mechanisms have been widely used in object detection models, enabling them to better focus on learning the feature information of particular objects. With a small increase in the size of the neural network, Hu et al. [23] embedded the SE module into a convolutional neural network to improve the representational power of the CNN. Jiang et al. [24] embedded an attention mechanism into a neural network and achieved similar detection performance with fewer parameters. These results illustrate the effectiveness of attention mechanisms. In practical production, detection models for pollination robots require the smallest model size possible. Adding an attention mechanism can improve the detection accuracy and recall of the model while only slightly increasing the number of model parameters.

Combining the above issues and methods, we chose Yolov5s as the base network. We then tried to enhance the feature learning ability for tomato flowers by adding a new feature fusion layer and the CBAM [25]. The purpose of adding a new feature fusion layer is to improve the model's adaptability to different scales of tomato flowers and to enhance the fusion of feature information between the upper and lower layers. The convolutional block attention module is a lightweight attention mechanism which enables the model to learn tomato images based on channels and spaces. It allows the model to focus on learning the tomato flowers themselves, while suppressing features other than the learning object. At the same time, the model proposed in this study can identify the different flowering phases of tomato flowers, which enhances the practicality of pollination robots performing pollination missions.

The remainder of this paper is organized as follows: Section 2 describes the data collection method and proposes a method to improve the tomato flower detection model. Section 3 describes the experimental results of the improved model. Section 4 discusses the feasibility of the proposed method, and Section 5 concludes the paper.

## 2. Materials and Methods

### *2.1. Data Collection and Augmentation*

#### 2.1.1. Data Collection

Our experiment was carried out in the Science and Technology Demonstration and Promotion Base of China Vegetable Quality Standard Center (Shouguang, Shandong) (Figure 1). The data sets were collected from the pollination robot independently developed by the National Intelligent Agricultural Equipment Research Center. The acquisition camera was a ZED2 HD camera, and the pollination robot core controller was configured with an i7-4700MQ@2.4 GHz CPU, 8 GB DDR3L memory, and a 500 GB SSD. As shown in Figure 1, the spacing between tomato plants was 0.2 m, and the width of the robot was 0.8 m. During the pollination robot operation, the robot used magnetic stripe navigation technology, and the travel speed on the track was set to 0.4 m/s.

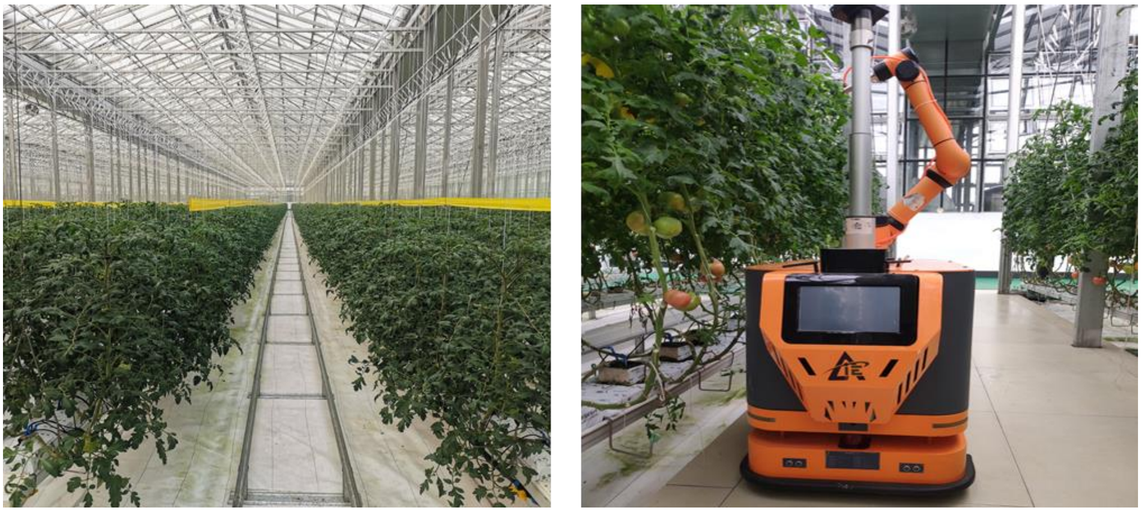

**Figure 1.** Experimental site and tomato pollination robot.

To ensure that the object detection model proposed in this paper can meet the actual production needs, the tomato flowering phase data set constructed for this paper contained a total of 1120 images. The data set was divided into a training set (896 images), a validation set (112 images), and a training set (112 images), according to the ratio of 8:1:1. Moreover, it included the tomato flowering phases of bud, full bloom, and first fruiting. The tomatoes photographed in the data set have the Latin name *Solanum lycopersicum* L., and their flowers are yellow at full bloom, with five radially spreading calyces and 1–1.5 cm long pedicels.

Meanwhile, to reduce the influence of light on the object detection model, the data set also contained images collected under different lighting conditions and at different times, as shown in Figure 2. The data samples were collected at three times—at 7:00 a.m., 12:00 p.m., and 4:00 p.m.—under two different weather conditions: sunny and overcast.

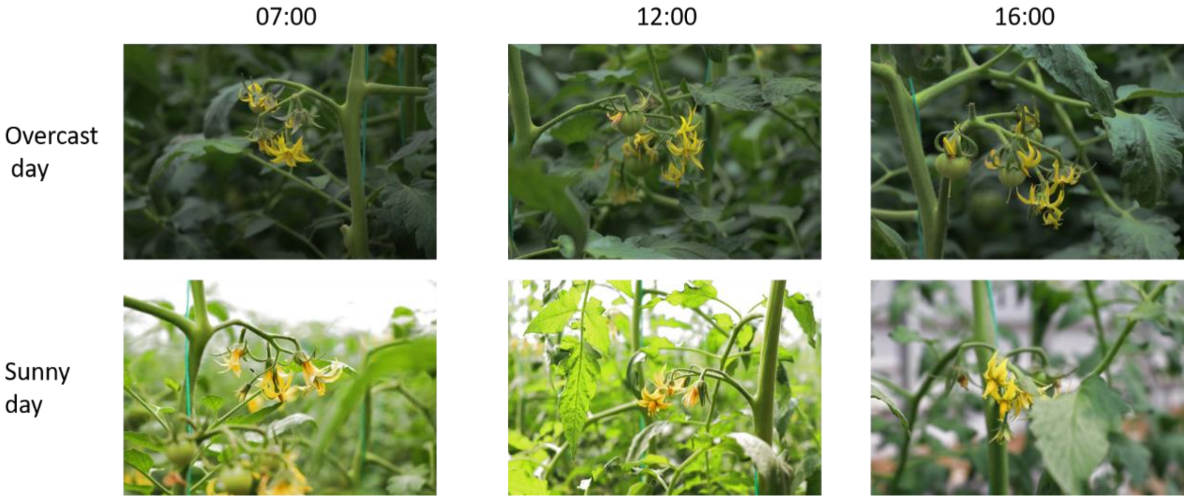

**Figure 2.** Examples of tomato flowers collected at different times and under different weather conditions.

In addition, we can clearly see from the figure: compared to photos taken on sunny days, photos taken on overcast days are darker, which increases the diversity of the dataset.

### 2.1.2. Data Augmentation

Before the data set was used for training, we performed data augmentation on it. As can be seen from Figure 3a–d, the original data set was first enhanced by color gamut adjustment, random scaling, horizontal mirroring, and panning. Then, as we can see from Figure 3e, every four images in the enhanced data set were randomly cropped and stitched into one image to form training data by using the mosaic method. According to the baseline, the four images were placed in the top-left, top-right, bottom-left, and bottom-right positions of the new large image. The advantage is that this greatly enriched the image background and increased the data diversity, and the four images stitched together served to increase the batch size. The number of objects was increased by calculating four images simultaneously, thus speeding up the convergence of the object detection model.

## *2.2. FlowerYolov5*

Due to the variety of flower objects and the complexity of the greenhouse background, we aimed to develop a robust network model to extract enough of the fine-grained features of flowers. The Yolov5 model adopts CSPDarknet53 as the backbone network and adds adaptive anchor frames to make the model training more applicable to training data sets. We used the Yolov5 object detection model as the basis, and fine-tuned the backbone and neck parts of the network to make it more suitable for detecting flower objects, with the aim of obtaining a higher recall and recognition accuracy.

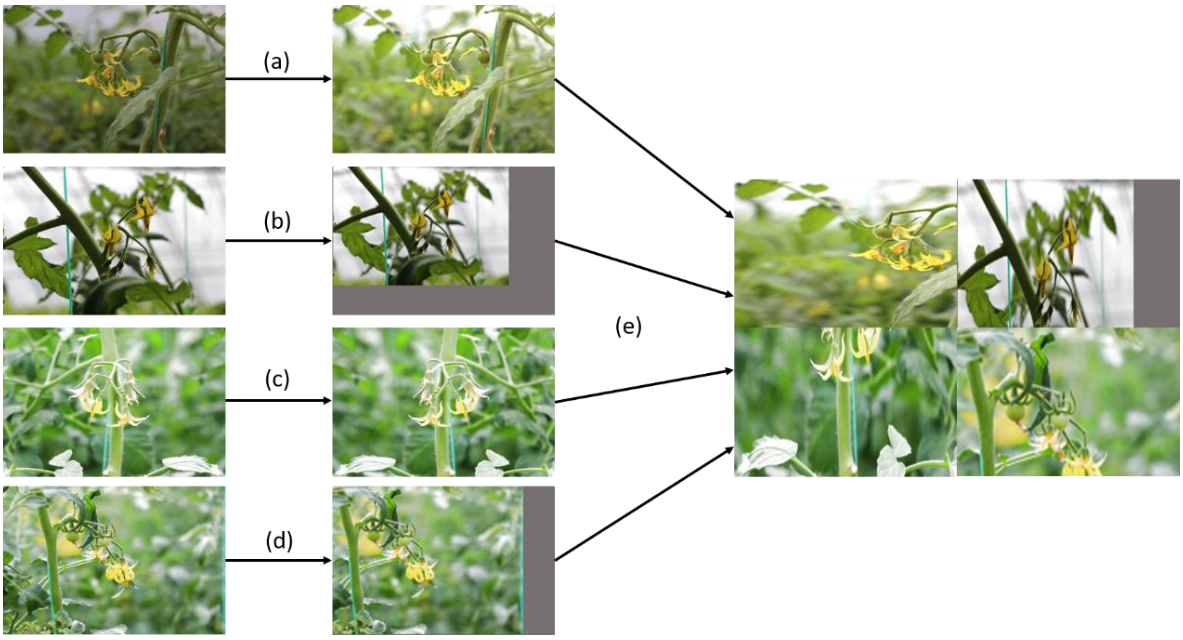

**Figure 3.** Data augmentation methods: (**a**) gamut adjustment; (**b**) random scaling; (**c**) horizontal mirroring; (**d**) panning; and (**e**) mosaic.

### 2.2.1. Backbone of Yolov5

Yolov5 adopts CSP Darknet [26] as its backbone, as shown in Figure 4, which consists of five CSP modules, one convolution module, and one spatial pyramid pooling module. The input image size is 640 × 640 × 3, which passes through a 7 × 7 convolution kernel with a stride of two and padding of one. Then, the feature map passes through five CSP blocks, each containing a 3 × 3 convolution kernel with a stride of two and padding of one, as well as a C3 block. The feature maps are divided into two parallel paths for propagation in the C3 block. Path 2 first passes through a convolutional layer with a 1 × 1 convolutional kernel and then passes through $n$ residual blocks containing two tiny paths. Path A passes through two convolutional layers, with kernels of size 1 × 1 and 3 × 3, respectively. The input of path B is directly summed with the output of path A, in order to obtain the output of each residual block. The $n$ residual blocks are used to calculate the output of path 2. Meanwhile, path 1 passes through a 1 × 1 convolutional kernel with a stride of one, which is then spliced with the output of path 2 and passed through another 1 × 1 convolutional kernel. After passing through five CSP blocks, the feature map in the backbone finally passes through the SPPF block, which contains three max-pooling layers and a 1 × 1 convolution kernel with a stride of one.

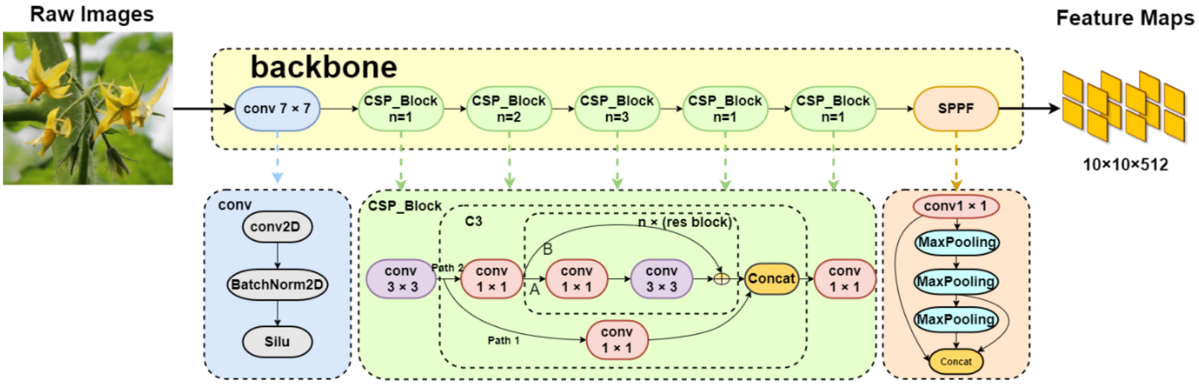

**Figure 4.** Local structure of the CSPNet backbone network.

### 2.2.2. Improvements to the Model

(1) Design the novel feature fusion layer

In this paper, we propose a method that adds a new feature fusion layer to the neck of Yolov5, in order to extract more fine-grained flower features and to reduce the feature information lost during the convolution from shallow to deep in the backbone.

There are three feature fusion layers in the original Yolov5 network model, with output feature vector dimensions of 80 × 80 × 128, 40 × 40 × 256, and 20 × 20 × 512, respectively. When the network is trained, the input images lose or blur feature information during multiple convolution and pooling operations in the network model, so it is necessary to strengthen the model's shallow and deep feature fusion ability. Therefore, we added a new feature fusion layer to the original Yolov5 model, as shown by the red dashed rectangular box in Figure 5. The new fusion feature layer is located at the bottom of the Neck section. It includes an Upsample block, two convolution blocks, a Concat block, and a C3 block. In the new feature fusion layer, it is not only necessary to pass the deep features through the Upsample block to the upper layer of the feature pyramid, but also to receive the shallow features passed from the upper end of the pyramid for feature fusion through the Concat block. The output feature vector dimensions of the new feature fusion layer are 10 × 10 × 512. Throughout the FlowerYolov5 structure, the new feature fusion layer has deep features as input. The deep-level features correspond to a larger receptive field, and the CNN is able to perform feature extraction from a more global perspective on the image, thus the network is able to acquire higher-level semantic information.

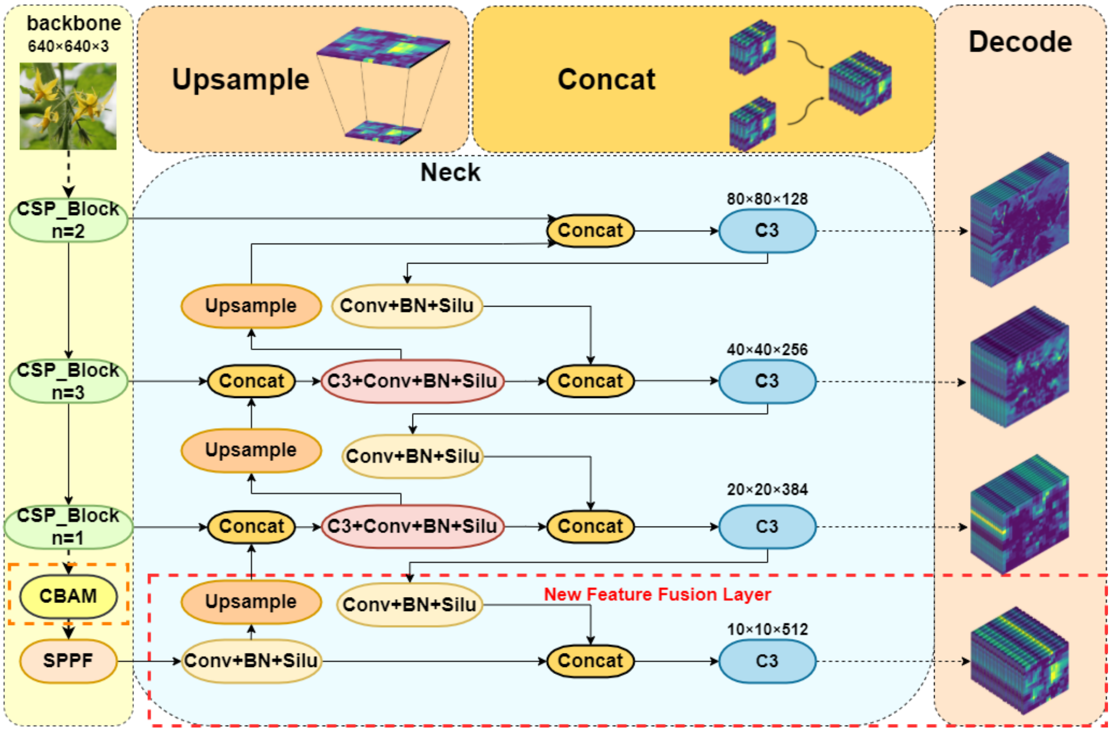

**Figure 5.** The architecture of FlowerYolov5.

(2) Insert the Attention Mechanism Module

In order to further enhance the recognition of flower objects in complex greenhouse environments, we introduced a convolutional block attention module (CBAM) [25] into the Yolov5 model, aiming to improve the expression of flower features (i.e., the model is expected to focus on learning flower features and suppress complex greenhouse background features during the training process). The CBAM module contains two independent sub-modules—namely, a channel attention module and a spatial attention module—which focus the training on flower images in terms of the channel and space, respectively, which not only reduces the required computation but also increases the accuracy of the model.

During training, the Channel Attention Module (CAM) focuses on learning the connections between the feature graph channels (i.e., those channels that play a role in the final output results of the network) in the CNN network. The input element graphs are first passed through the maximum pooling layer and the average pooling layer based on width and height, respectively, and then the results are fed into the MLP. The output features from the MLP are then subjected to summation operations and sigmoid activation to output the CAM feature map, which is multiplied by the input feature map to generate the input features required by the SAM module.

The spatial attention module (SAM) is focused on learning the features of the regions of interest of the feature map flower objects in the CNN network (i.e., it focuses on the location information that plays a role in the final output result of the network). The input feature map is concated after channel-based maximum and average pooling layers, then is downscaled to a channel by the convolution layer and subjected to sigmoid activation. The SAM feature map is the output, and this feature map and the input feature map are multiplied to output the final generated feature map.

The CAM and SAM are combined in serial order to form the CBAM; the structure is shown in Figure 6. After the above two modules, the network focuses on learning the feature information of the feature map on the channel and spatial objects, and can better extract the fine-grained feature information.

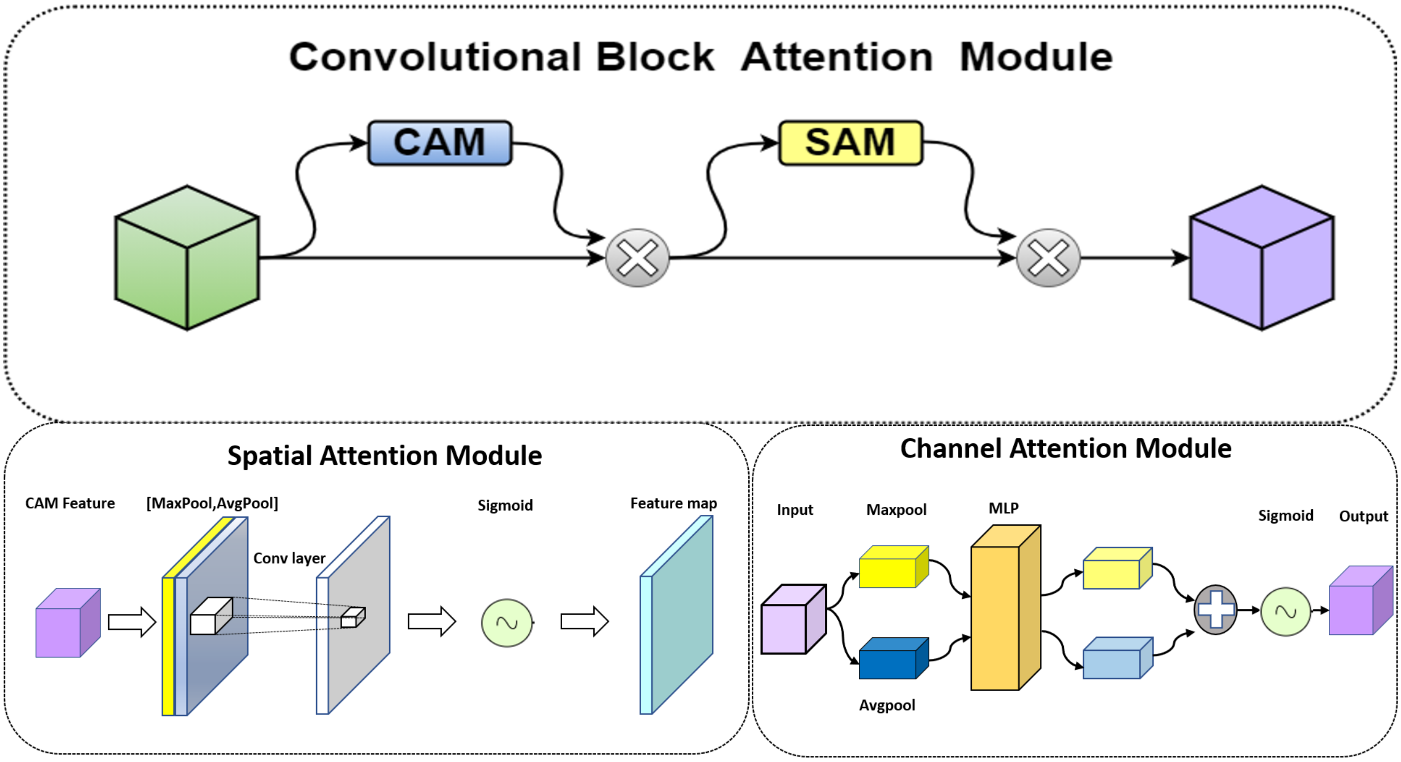

**Figure 6.** CBAM structure.

We embedded CBAM into the backbone after the last CSP block (see the orange dashed rectangle in Figure 5 for its position).

## 2.3. Bounding Box Regression and Loss Function

The IoU (intersection over union) is a criterion to judge the accuracy of object detection tasks. It is the ratio of the intersection and concatenation of the output-predicted bounding boxes of an object detection model and manually labeled real bounding boxes, which reflects the similarity between the predicted and real bounding boxes. The value of the IoU is in the range of [0, 1], and its calculation formula is as follows:

$$\mathrm{IoU} = \frac{areaPred \cap Truth}{areaPred \cup Truth}. \tag{1}$$

The IoU loss is used to constrain the size and position of the predicted bounding boxes, in order to gradually regress them to the manually labeled true bounding boxes, as follows:

$$L_{\mathrm{IoU}} = -ln\mathrm{IoU}. \tag{2}$$

However, when the predicted and real bounding boxes do not intersect, the IoU is 0 and the prediction bounding box cannot converge properly, resulting in deviation in the position and size of the prediction box. Therefore, some studies [27,28] have proposed three new methods to calculate the predicted bounding box loss: GIoU, DIoU, and CIoU [29]. The CIoU considers three geometric parameters: the overlap between two prediction boxes, the centroid distance, and the aspect ratio. Compared to the GIoU and DIoU, the CIoU has a faster convergence speed and a higher regression accuracy. Therefore, in this paper, the CIoU was used as the evaluation index for the predicted bounding boxes, and its calculation formula is as follows:

$$\mathrm{CIoU} = \mathrm{IoU} - \left( \frac{\rho^2\left(b^{Truth}, b^{Pred}\right)}{c^2} + \alpha v \right), \tag{3}$$

$$v = \frac{4}{\pi^2}(arctan\frac{w^{Truth}}{h^{Truth}} - arctan\frac{w}{h})^2, \tag{4}$$

$$\alpha = \frac{v}{(1 - \mathrm{IoU}) + v} \tag{5}$$

where $\rho$ is the Euclidean distance between the center points of the real and predicted bounding boxes, $b$ is the center point of the bounding box, $c$ is the length of the diagonal of the minimum outer rectangle of the real and predicted bounding boxes, and $w$ and $h$ are the length and width of the bounding box, respectively.

FlowerYolov5 contains three kinds of loss functions: box_*loss*, *cls_loss*, and *obj_loss*. box_*loss* is the $L_{\mathrm{CIoU}}$, expressed as:

$$L_{\mathrm{CIoU}} = 1 - \mathrm{CIoU} \tag{6}$$

*Obj_loss* is the confidence loss, which can be expressed as:

$$\begin{gathered} L_{obj} = \lambda_{obj} \textstyle\sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{i,j}^{obj} [\hat{C}_i^j log(C_i^j) + (1 - \hat{C}_i^j) log(1 - C_i^j)] + \\ \lambda_{noobj} \textstyle\sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{i,j}^{noobj} [\hat{C}_i^j log(C_i^j) + (1 - \hat{C}_i^j) log(1 - C_i^j)], \end{gathered} \tag{7}$$

where $I_{i,j}^{noobj}$ indicates that the $j$th bounding box in the $i$th grid takes a value of 0 if there is a detection object, and takes a value of 1 if there is not; $I_{i,j}^{obj}$ means that the $j$th bounding box in the $i$th grid takes a value of 1 if there is a detection object, and otherwise takes a value of 0; $S$ denotes the grid size; $C_i^j$ denotes the confidence score of the true bounding box; $\hat{C}_i^j$ denotes the confidence score of the predicted bounding box; and $B$ denotes the number of a priori boxes in each grid.

*cls_loss* is the category loss, which can be expressed as:

$$L_{cls} = \textstyle\sum_{i=0}^{S^2} I_{i,j}^{obj} \sum_{C \in classes} [\hat{p}_i(c) log(p_i(c)) + (1 - \hat{p}_i(c)) log(1 - p_i(c))], \tag{8}$$

where $c$ denotes the class of the detected object, $p_i(c)$ denotes the probability that the actual detected object is $c$, and $\hat{p}_i(c)$ denotes the probability that the predicted object is $c$.

The above three loss functions together constitute the total loss function of the FlowerYolov5 objective detection model, which can be expressed as:

$$LOSS = L_{\mathrm{CIoU}} + L_{obj} + L_{cls}. \tag{9}$$

## 3. Results

### 3.1. Evaluation Indicators

In order to be able to evaluate the performance of the models objectively, four evaluation metrics—including *Precision*, *Recall*, *mAP*, and $F_1$—were used in this experiment to comprehensively evaluate the classification performance of each model, which can be expressed as follows:

$$Precision = \frac{TP}{TP + FN} * 100\%, \tag{10}$$

$$Recall = \frac{TP}{TP + FP} * 100\%, \tag{11}$$

$$mAP = \frac{1}{C}\sum_{k=i}^{N} Precision(k)Recall(k), \tag{12}$$

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{13}$$

where *TP* denotes the number of positive samples correctly classified as positive samples, *FN* denotes the number of negative samples incorrectly classified as positive samples, *FP* denotes the number of positive samples incorrectly classified as negative samples, and $C$ denotes the number of detected object classes.

The *mAP* index is the average value of *AP* over each category recognized by the model, which can provide a more comprehensive measure of a model's ability to recognize each category of objects: the higher the *mAP* value, the higher the recognition accuracy of the model and the lower the misdetection rate. The $F_1$ score balances both the precision and recall of the classification model. Additionally, it can be considered as a harmonic mean of accuracy and recall, whose maximum value is 1 and minimum value is 0.

In addition, we also discuss the robustness of the model. The robustness of neural networks can be understood as the stability of the model to changes in the data. When the input data or information undergoes limited change, the model can still maintain stable output. The more robust the model is, the more stable its output performance will be in case of input data disturbance.

### 3.2. Test Training Platform

The hardware configuration used for network training in this paper consisted of an Intel$^{(R)}$ Core$^{(TM)}$ i9-11900K@3.50 GHz CPU, 64 GB running memory, a 2 TB HDD, and a 24 GB NVIDIA GeForce RTX 3090 GPU, while the experimental environment was Ubuntu, under which the Pytorch deep learning framework was used. The program was written on the PyCharm platform using the Python 3.8 language.

The proposed FlowerYolov5 model receives 640 × 640 pixel images as input, uses stochastic gradient descent (SGD) as the optimizer to optimize the network parameters, with an initial learning rate (lr0) of 0.01, initial learning rate momentum of 0.937, weight decay of 0.0005, a batch size of eight, and each model is set to train for 500 epochs.

### 3.3. Experimental Comparison

Table 1 lists the model performance comparisons of FlowerYolov5, Yolov3 [27], Yolov4 [28], Yolov5s, Yolov5m, and Yolov5x. It can be seen, from Table 1, that all of the metrics of Yolov5s were substantially improved after introducing the CBAM attention mechanism and adding a feature fusion layer. Compared to the original Yolov5s model, the *precision* of FlowerYolov5 was improved by 9.7%, the *recall* was improved by 3.3%, and the $F_1$ score was improved by 6.6%.

**Table 1.** Comparison results of the different models for tomato flowers.

| | **Backbone** | ***Precision*** | ***Recall*** | $F_1$ | ***mAP*** | **Mbyte** |
|---|---|---|---|---|---|---|
| Yolov3 | Darknet-53 | 0.894 | 0.798 | 0.843 | 0.874 | 63.5 |
| Yolov4 | CSPDarknet-53 | 0.895 | 0.838 | 0.866 | 0.89 | 65.5 |
| Yolov5s | CSPDarknet-53 | 0.802 | 0.867 | 0.833 | 0.864 | 13.7 |
| Yolov5m | CSPDarknet-53 | 0.880 | 0.847 | 0.863 | 0.891 | 40.2 |
| Yolov5l | CSPDarknet-53 | 0.833 | 0.910 | 0.869 | 0.911 | 88.5 |
| FlowerYolov5 | CSPDarknet-53 | 0.899 | 0.900 | 0.899 | 0.942 | 23.9 |

To judge the recognition accuracy of the model proposed in this paper, we compared the *mAP* index of the model to those of the other classical object detection models. Yolov5x had a larger model, meaning that it cannot satisfy the lightweight requirements of pollination robot detection; therefore, it was reasonably abandoned. Figure 7 shows the *mAP* curves of the proposed FlowerYolov5 model, as well as those of the four other classical object detection models. It can be seen that the total *mAP* value for each model increased with the number of epochs, stabilizing when they reached approximately 300 epochs. The convergence speed of FlowerYolov5 was the fastest, and its *mAP* value was the highest (94.2%). The *mAP* was improved by 6.8%, 5.2%, 7.8%, and 5.1%, respectively, compared to the four other classical object detection networks.

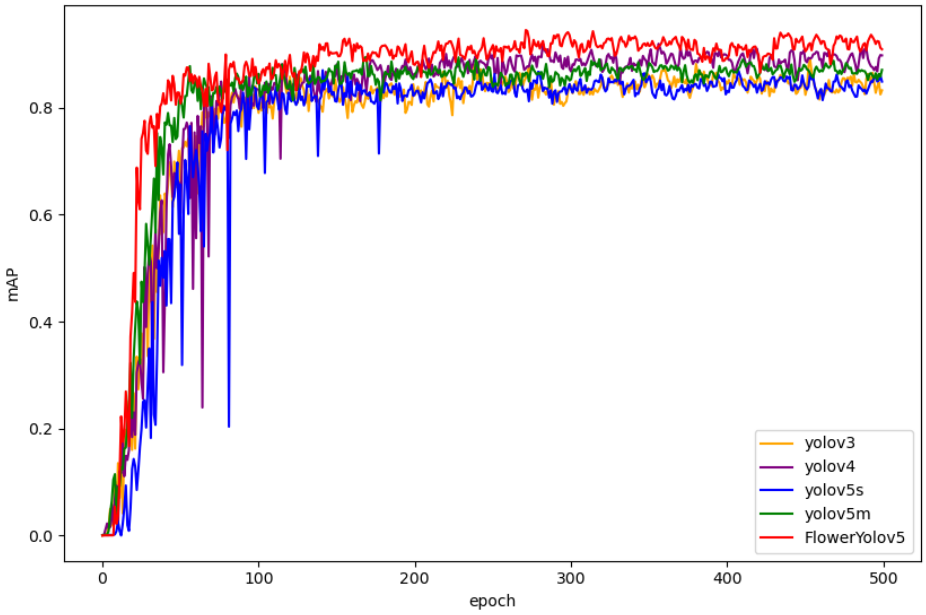

**Figure 7.** *mAP* curves of the five object detection models.

The variation curves for the total loss function values of FlowerYolov5 and the other object detection networks are given in Figure 8, further validating the detection advantages of the FlowerYolov5 model. The total loss function of the model can be used to measure the model's performance by determining the agreement between the real and predicted object frames. The lower the loss value, the better the detection performance of the model and the more consistent the true and predicted objects obtained. From Figure 8, we can see that the overall trend of the loss function for all models decreased until the number of epochs reached 400, and the loss function tended to stabilize and converge after this point. The total loss value of the FlowerYolov5 model was the smallest, reaching approximately 0.031.

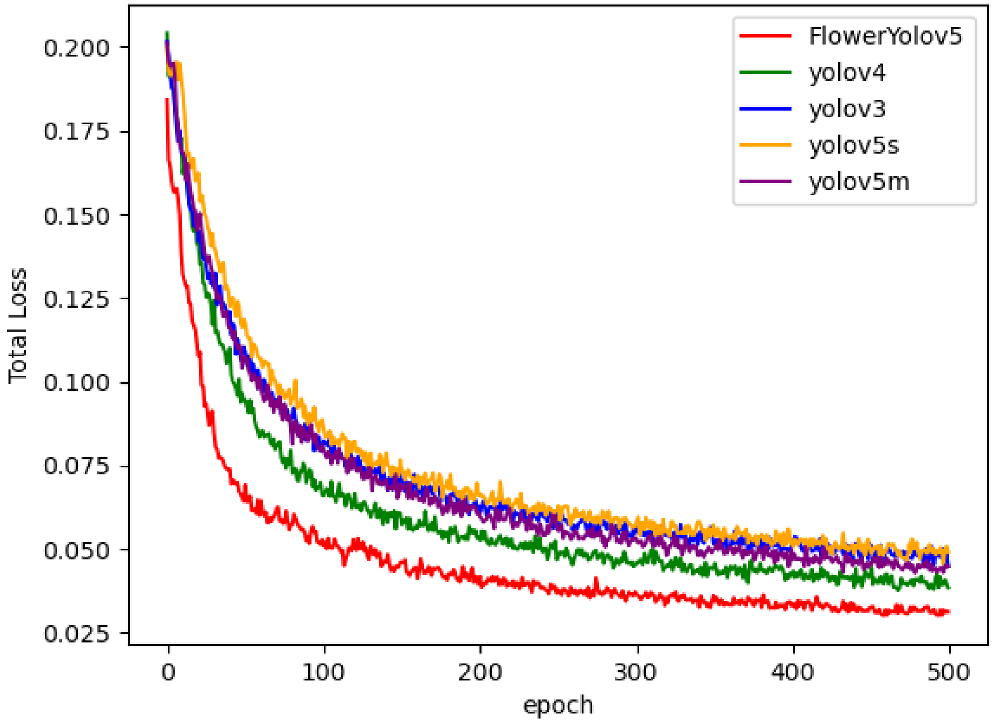

**Figure 8.** Total loss curves of the five object detection models.

Combined with the parameter calculation and detection accuracy values, the overall performance of FlowerYolov5 indicates that it is more suitable for embedding into pollination robots to perform intelligent pollination tasks in complex greenhouse environments.

## 4. Discussion

In the remainder of this paper, we investigate the impact of introducing the CBAM at different locations in the Yolov5s model on its accuracy. Meanwhile, we discuss the impact of adding the feature fusion layer on the performance of the original network.

We embedded the CBAM after all of the CSP blocks in the backbone, all the CSP blocks in the neck, all of the CSP blocks in the Yolov5s model, and the last CSP block in the backbone, respectively. After this, we compared the four models. As shown in Figure 9, from the $F_1$, *recall*, and *precision* histograms of the four different models in the experiment, we can observe that, regardless of the position in which the CBAM was introduced, the *precision*, *recall*, and $F_1$ values were significantly improved, compared to the original network. After embedding CBAM into all C3 modules in the neck, the $F_1$ score improved by 2.9%; after embedding CBAM into all C3 modules in the backbone, the $F_1$ score improved by 1.1%; after embedding CBAM into all C3 modules in the neck and the backbone, the $F_1$ score improved by 2.9%; finally, the most apparent improvement was achieved after embedding the CBAM after the last CSP block in the backbone, with a 4.5% improvement in *precision*, a 2.9% improvement in *recall*, and a 3.7% improvement in the $F_1$ score.

Combining the above experimental results, we can draw the following conclusions: The introduction of the CBAM can greatly enhance the performance of the model, reduce the misdetection rate, and increase the accuracy of the model. Among them, the highest accuracy and the best detection performance were achieved by embedding it after the last CSP block module of the backbone.

In order to verify whether the performance of the network was improved after adding the feature fusion layer, we conducted the following fusion experiments, as detailed in Table 2, which provides a comparison of the different flowering phase APs of the Yolov5s model after adding the feature fusion layer, the original Yolov5s, and Yolov5s after introducing the CBAM module to the FlowerYolov5 model proposed in this paper. As can be seen from Table 2, compared to the original Yolov5s network, adding a feature fusion layer improved the AP of the network model substantially in the bud and early fruit phases by 2% and 11.7%, respectively. This indicates that adding a feature fusion layer allows the model to better obtain the fine-grained features of flowers, resulting in a higher accuracy for the detection of multiple flower phases.

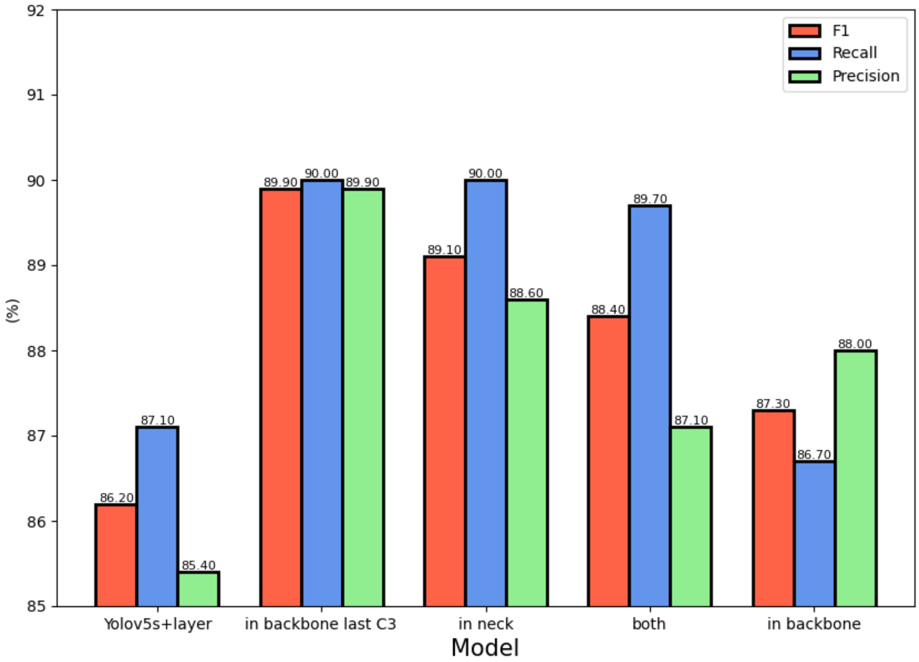

**Figure 9.** Impact of adding the CBAM in different locations on the model performance indices.

**Table 2.** Comparison results of model performance using different improvement methods.

| | Bud Phase (AP) | Bloom Phase (AP) | Early Fruit Phase (AP) |
|---|---|---|---|
| Yolov5s | 79.5% | 96.6% | 83.1% |
| Yolov5s + fusion layer | 81.5% | 94.8% | 94.8% |
| Yolov5s + CBAM | 84.9% | 97.7% | 85.3% |
| FlowerYolov5 | 90.5% | 97.7% | 94.9% |

We further visualized the results to illustrate the improvement in the model's recognition accuracy and recall after adding a feature fusion layer and the CBAM. Figure 10 compares the prediction results obtained by the four models: Yolov5s, Yolov5s with the addition of a feature fusion layer, Yolov5s with the addition of the CBAM, and FlowerYolov5. As can be seen from Figure 10, after adding a feature fusion layer or the CBAM into the model, both the confidence level and the *recall* were higher than that of the original Yolov5s model. Among the models, FlowerYolov5 had an overall higher confidence level and was able to detect the flowers missed by the other three models. As can be seen from Figure 10e, FlowerYolov5 was able to detect one or two more flowers compared to the original Yolov5. This illustrates the effectiveness of adding feature fusion layers and the CBAM.

In addition, we can see from Figure 10 that the greenhouse background is quite complex and there is occlusion between flowers. In Figure 10b–d, the models missed the detection of several flowers, while FlowerYolov5 exhibits a more robust recognition performance in Figure 10e.

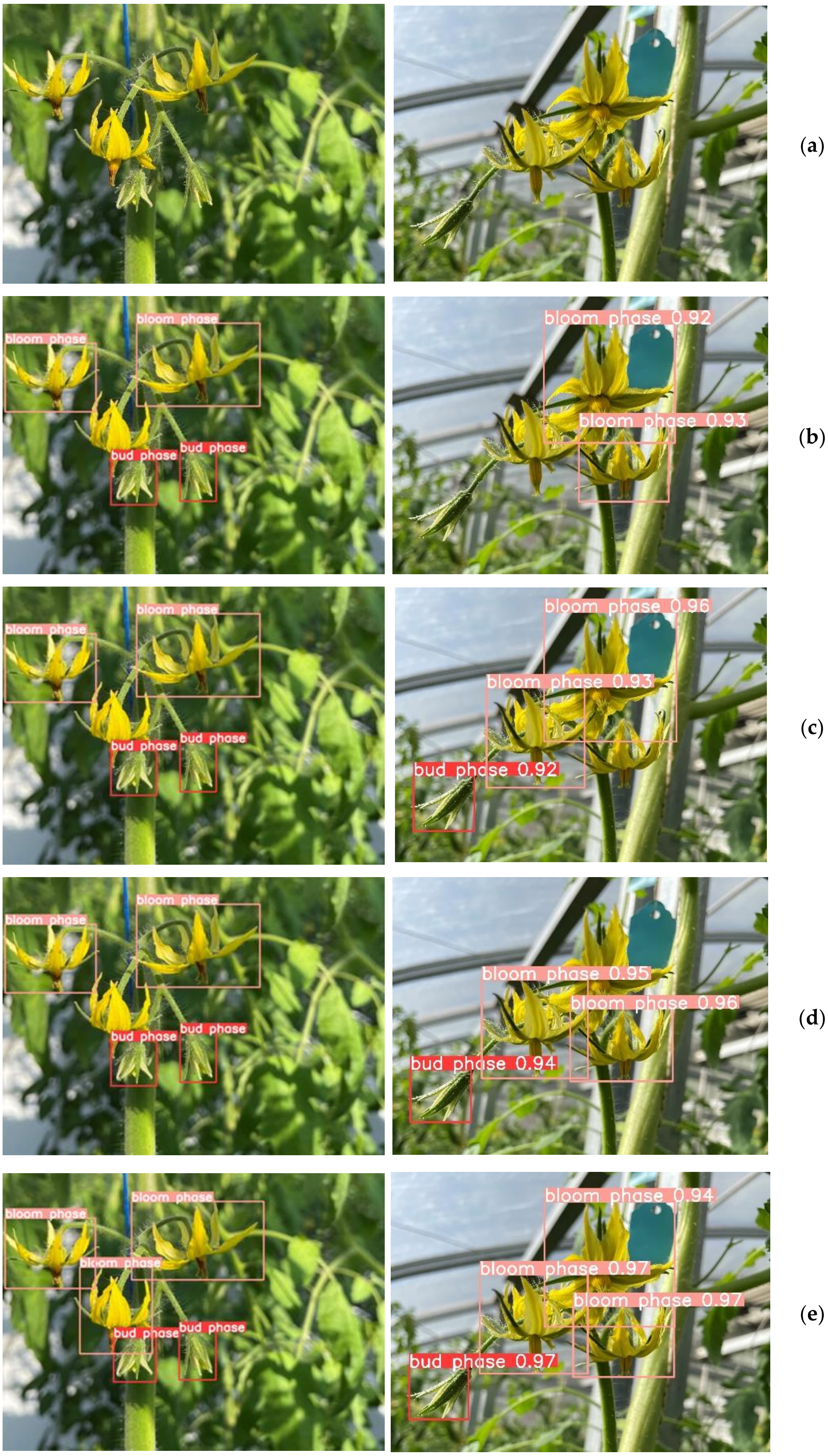

**Figure 10.** Recognition results of the comparison between models for different phases: (**a**) original images; (**b**) Yolov5s; (**c**) Yolov5s + fusion layer; (**d**) Yolov5s + CBAM; and (**e**) FlowerYolov5.

To demonstrate that the CBAM and extra feature fusion layer can effectively enhance the feature extraction ability, we further compared the convolutional outputs of the original Yolov5, Yolov5 with the addition of CBAM, and FlowerYolov5. Then, we visualized the output feature vector before the final convolutional layers from models in order to discern the tomato flower feature extraction ability of the models. Figure 11a shows the original image, while Figure 11b–d demonstrate the results of feature visualization for the original Yolov5, Yolov5 with the CBAM added, and FlowerYolov5, respectively. In this figure, we can see two different flowering classes; namely, the full bloom and bud phases. Each point in the feature map is a representation of the activation value, with red or yellow highlighting relatively large activation values and blue indicating lower activation values, representing the background. The energy distribution visualization for the feature maps demonstrates that Yolov5 with the addition of the CBAM can effectively focus on activating discriminative regions while ignoring the complicated and messy backgrounds. From Figure 11b–d, we can see that most of the activation points were transferred from the surroundings to the areas of the flowers, highlighting these areas instead of cluttering the background. This shows that the model can focus more on extracting flower color and texture features. We can observe that FlowerYolov5 successfully extracted features that were easily lost in the deep convolutional layer. The effectiveness of the FlowerYolov5 network in fine-grained flowering phase identification was thus demonstrated.

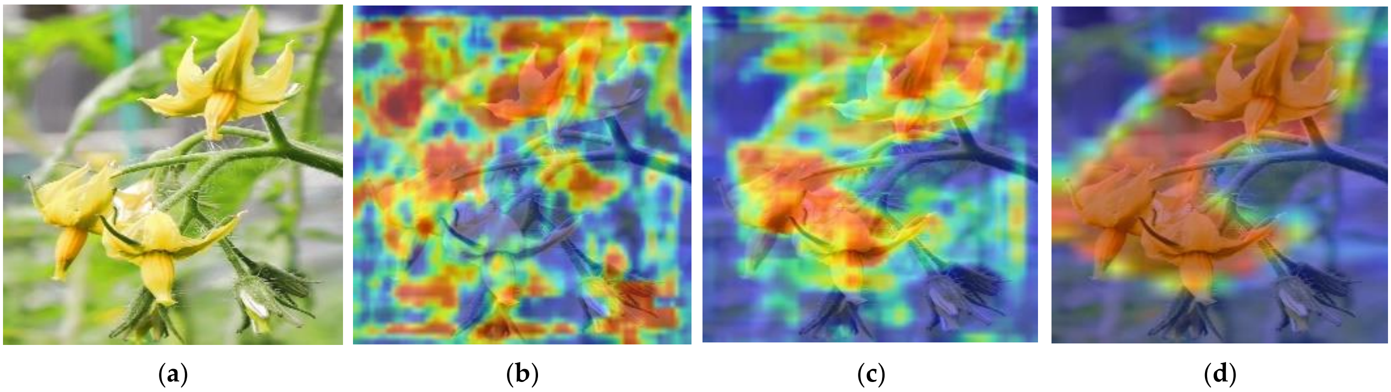

**Figure 11.** Energy distribution visualization of feature maps before and after adding a feature fusion layer and the CBAM: (**a**) original image; (**b**) original Yolov5; (**c**) Yolov5 with the CBAM added; and (**d**) FlowerYolov5.

## 5. Conclusions

The detection of the tomato flowering phase and flower identification in intelligent greenhouses are of great significance for improving the yield and quality of tomatoes. In this paper, we proposed an object detection method based on the improved Yolov5, which improves the recognition accuracy for flowers and can achieve the accurate identification of different flowering phases. As for the modification method, a feature fusion layer was added and embedded into the output end of Yolov5 in order to reduce the amount of semantic information lost in the image during the convolution process. Furthermore, the CBAM was added to the backbone network in order to improve the detection accuracy of floral objects. According to the experimental results, the following conclusions can be drawn:

(1) The model performance verification experiment showed that FlowerYolov5 achieved a better performance, with 90.5% AP for the bud phase, 97.7% AP for the bloom phase, and 94.9% AP for the early fruit phase. In general, the mean average *precision* reached 94.2%, which is 7.6% better than that of the original Yolov5 network. Therefore, FlowerYolov5 can more accurately identify and classify different flowering phases, and provides a technical reference for precise identification by pollination robots.

(2) A comparison of the detection results showed that the performance of FlowerYolov5 was generally better than that of Yolo series networks. The previous problem related to undetected flowers was improved.

Validation experiments and analyses demonstrated the better efficiency and robustness of the proposed method, which can meet the practical demands of production management in different intelligent greenhouse applications. In the future, the approach proposed in this paper can be combined with other advanced information technologies, such as fractional-order bidirectional associative memory neural network [30–34], artificial intelligence and big data mining algorithms [35–37], to the study of pattern recognition problems for linear and non-linear systems, and can be applied to other fields such as time-series forecasting and engineering application systems [38–40].

**Author Contributions:** T.X., investigation and writing—review and editing. X.Q., conceptualization, methodology, software, and writing—original draft. S.L., investigation, data curation, and supervision. Y.G., writing—review and editing and software. Y.Z., investigation, data curation, and supervision. Z.L., data curation. J.D., investigation. X.Y., investigation and data curation. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was financially supported by Beijing Municipal Science and Technology Project (Z211100004621006), National Natural Science Foundation of China (62006008), Youth Research Fund of Beijing Academy of Agriculture and Forestry Sciences (QNJJ202027) and Science & Technology Innovation 2030 Project (2021ZD0113602).

**Data Availability Statement:** The data used to support the findings of this study are available from the corresponding author upon request.

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

## References

1. Kong, J.; Wang, H.; Wang, X.; Jin, X.; Fang, X.; Lin, S. Multi-stream Hybrid Architecture Based on Cross-level Fusion Strategy for Fine-grained Crop Species Recognition in Precision Agriculture. *Comput. Electron. Agric.* **2021**, *185*, 106134. [CrossRef]
2. Zheng, Y.; Kong, J.; Jin, X.; Wang, X.; Su, T.; Zuo, M. Crop Deep: The Crop Vision Dataset for Deep-learning-based Classification and Detection in Precision Agriculture. *Sensors* **2019**, *19*, 1058. [CrossRef] [PubMed]
3. Kong, J.; Yang, C.; Wang, J.; Wang, X.; Zuo, M.; Jin, X.; Lin, S. Deep-stacking network approach by multisource data mining for hazardous risk identification in IoT-based intelligent food management systems. *Comput. Intell. Neurosci.* **2021**, *2021*, 1194565. [CrossRef] [PubMed]
4. Tong, Y.; Yu, L.; Li, S.; Liu, J.; Qin, H.; Li, W. Polynomial Fitting Algorithm Based on Neural Network. *ASP Trans. Pattern Recognit. Intell. Syst.* **2021**, *1*, 32–39. [CrossRef]
5. Ning, X.; Wang, Y.; Tian, W.; Liu, L.; Cai, W. A Biomimetic Covering Learning Method Based on Principle of Homology Continuity. *ASP Trans. Pattern Recognit. Intell. Syst.* **2021**, *1*, 9–16. [CrossRef]
6. Ahmad, A.A. Automated Flower Species Detection and Recognition from Digital Images. *Int. J. Comput. Sci. Net.* **2017**, *17*, 144–151.
7. Aleya, K.F. Automated damaged flower detection using image processing. *J. Glob. Res. Comput. Sci.* **2013**, *4*, 21–24.
8. Dorj, U.; Lee, M.; Diyan-ul-Imaa. A New Method for Tangerine Tree Flower Recognition. *Commun. Comput. Inf. Sci.* **2012**, *353*, 49–56.
9. Jin, X.-B.; Zheng, W.-Z.; Kong, J.-L.; Wang, X.-Y.; Bai, Y.-T.; Su, T.-L.; Lin, S. Deep-learning Forecasting Method for Electric Power Load Via Attention-based encoder-decoder with Bayesian Optimization. *Energies* **2021**, *14*, 1596. [CrossRef]
10. Jin, X.-B.; Zheng, W.-Z.; Kong, J.-L.; Wang, X.-Y.; Bai, Y.-T.; Su, T.-L.; Lin, S. Probability Fusion Decision Framework of Multiple Deep Neural Networks for Fine-grained Visual Classification. *IEEE Access* **2019**, *7*, 122740–122757.
11. Chen, Y.; Lee, W.S.; Gan, H.; Peres, N.; Fraisse, C.; Zhang, Y.; He, Y. Strawberry Yield Prediction Based on a Deep Neural Network Using High-Resolution Aerial Orthoimages. *Remote Sens.* **2019**, *11*, 1584. [CrossRef]
12. Sun, J.; He, X.; Ge, X.; Wu, X.; Shen, J.; Song, Y. Detection of Key Organs in Tomato Based on Deep Migration Learning in a Complex Background. *Agriculture* **2018**, *8*, 196. [CrossRef]
13. Sun, K.; Wang, X.; Liu, S.; Liu, C. Apple, peach, and pear flower detection using semantic segmentation network and shape constraint level set. *Comput. Electron. Agric.* **2021**, *185*, 106150. [CrossRef]
14. Saad, W.H.M.; Karim, S.A.A.; Razak, M.S.J.A.; Radzi, S.A.; Yussof, Z.M. Classification and detection of chili and its flower using deep learning approach. *J. Phys. Conf. Ser.* **2020**, *1502*, 012055. [CrossRef]
15. Chu, Z.; Hu, M.; Chen, X. Robotic grasp detection using a novel two-stage approach. *ASP Trans. Internet Things* **2021**, *1*, 19–29. [CrossRef]

16. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Washington, DC, USA, 7–13 December 2015; pp. 1440–1448.
17. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *42*, 386–397. [CrossRef]
18. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
19. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; IEEE: New York, NY, USA, 2017; pp. 6517–6525.
20. Huang, Z.; Zhang, P.; Liu, R.; Li, D. Immature Apple Detection Method Based on Improved Yolov3. *ASP Trans. Internet Things* **2021**, *1*, 9–13. [CrossRef]
21. Tian, M.; Chen, H.; Wang, Q. Detection and Recognition of Flower Image Based on SSD network in Video Stream. *J. Phys. Conf. Ser.* **2019**, *1237*, 032045. [CrossRef]
22. Cheng, Z.; Zhang, F. Flower End-to-End Detection Based on YOLOv4 Using a Mobile Device. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 8870649. [CrossRef]
23. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7132–7141.
24. Jiang, Y.; Chen, L.; Zhang, H.; Xiao, X. Breast cancer histopathological image classification using convolutional neural networks with small SE-ResNet module. *PLoS ONE* **2019**, *14*, e0214587. [CrossRef]
25. Woo, S.; Park, J.; Lee, J.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; Springer: Cham, Switzerland, 2018.
26. Wang, C.; Liao, H.M.; Wu, Y.; Chen, P.; Hsieh, J.; Yeh, I. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 16–18 June 2020; IEEE: New York, NY, USA, 2020.
27. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
28. Bochkovskiy, A.; Wang, C.Y.; Liao, H. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
29. Zheng, Z.; Wang, P.; Ren, D.; Liu, W.; Ye, R.; Hu, Q.; Zuo, W. Enhancing Geometric Factors in Model Learning and Inference for Object Detection and Instance Segmentation. *arXiv* **2019**, arXiv:2005.03572. [CrossRef] [PubMed]
30. Huang, C.; Wang, J.; Chen, X.; Cao, J. Bifurcations in a fractional-order BAM neural network with four different delays. *Neural Netw.* **2021**, *141*, 344–354. [CrossRef] [PubMed]
31. Huang, C.; Liu, H.; Shi, X.; Chen, X.; Xiao, M.; Wang, Z.; Cao, J. Bifurcations in a fractional-order neural network with multiple leakage delays. *Neural Netw.* **2020**, *131*, 115–126. [CrossRef]
32. Xu, C.; Liu, Z.; Aouiti, C.; Li, P.; Yao, L.; Yan, J. New exploration on bifurcation for fractional-order quaternion-valued neural networks involving leakage delays. *Cogn Neurodyn.* **2022**, *16*, 1233–1248. [CrossRef]
33. Huang, C.; Li, Z.; Ding, D.; Cao, J. Bifurcation analysis in a delayed fractional neural network involving self-connection. *Neurocomputing* **2018**, *314*, 186–197. [CrossRef]
34. Huang, C.; Zhao, X.; Wang, X.; Wang, Z.; Xiao, M.; Cao, J. Disparate delays-induced bifurcations in a fractional-order neural network. *J. Frankl. Inst.* **2019**, *356*, 2825–2846. [CrossRef]
35. Kong, J.; Yang, C.; Lin, S.; Ma, K.; Xiao, Y.; Zhu, Q. A Graph-related high-order neural network architecture via feature aggregation enhancement for identify application of diseases and pests. *Comput. Intell. Neurosci.* **2022**, *2022*, 4391491. [CrossRef]
36. Kong, J.; Wang, H.; Yang, C.; Jin, X.; Zuo, M.; Zhang, X. A Spatial Feature-Enhanced Attention Neural Network with High-Order Pooling Representation for Application in Pest and Disease Recognition. *Agriculture* **2022**, *12*, 500. [CrossRef]
37. Jin, X.; Zheng, W.; Kong, J.; Wang, X.; Zuo, M.; Zhang, Q.; Lin, S. Deep-Learning Temporal Predictor via Bidirectional Self-Attentive Encoder–Decoder Framework for IOT-Based Environmental Sensing in Intelligent Greenhouse. *Agriculture* **2021**, *11*, 802. [CrossRef]
38. Jin, X.-B.; Gong, W.-T.; Kong, J.-L.; Bai, Y.-T.; Su, T.-L. PFVAE: A Planar Flow-Based Variational Auto-Encoder Prediction Model for Time Series Data. *Mathematics* **2022**, *10*, 610. [CrossRef]
39. Jin, X.; Zhang, J.; Kong, J.; Su, T.; Bai, Y. A Reversible Automatic Selection Normalization (RASN) Deep Network for Predicting in the Smart Agriculture System. *Agronomy* **2022**, *12*, 591. [CrossRef]
40. Jin, X.; Gong, W.; Kong, J.; Bai, Y.; Su, T. A Variational Bayesian Deep Network with Data Self-Screening Layer for Massive Time-Series Data Forecasting. *Entropy* **2022**, *24*, 335. [CrossRef]