agriculture

MDPI

*Article*

# A Novel Lightweight Grape Detection Method

**Shuzhi Su [1,2,*], Runbin Chen [1], Xianjin Fang [1,2], Yanmin Zhu [3], Tian Zhang [1] and Zengbao Xu [1]**

[1] School of Computer Science and Engineering, Anhui University of Science & Technology, Huainan 232001, China
[2] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230031, China
[3] School of Computer Mechanical Engineering, Anhui University of Science & Technology, Huainan 232001, China
* Correspondence: szsu@aust.edu.cn

**Abstract:** This study proposes a novel lightweight grape detection method. First, the backbone network of our method is Uniformer, which captures long-range dependencies and further improves the feature extraction capability. Then, a Bi-directional Path Aggregation Network (BiPANet) is presented to fuse low-resolution feature maps with strong semantic information and high-resolution feature maps with detailed information. BiPANet is constructed by introducing a novel cross-layer feature enhancement strategy into the Path Aggregation Network, which fuses more feature information with a significant reduction in the number of parameters and computational complexity. To improve the localization accuracy of the optimal bounding boxes, a Reposition Non-Maximum Suppression (R-NMS) algorithm is further proposed in post-processing. The algorithm performs repositioning operations on the optimal bounding boxes by using the position information of the bounding boxes around the optimal bounding boxes. Experiments on the WGISD show that our method achieves 87.7% mAP, 88.6% precision, 78.3% recall, 83.1% F1 score, and 46 FPS. Compared with YOLOx, YOLOv4, YOLOv3, Faster R-CNN, SSD, and RetinaNet, the mAP of our method is increased by 0.8%, 1.7%, 3.5%, 21.4%, 2.5%, and 13.3%, respectively, and the FPS of our method is increased by 2, 8, 2, 26, 0, and 10, respectively. Similar conclusions can be obtained on another grape dataset. Encouraging experimental results show that our method can achieve better performance than other recognized detection methods in the grape detection tasks.

**Keywords:** grape detection; convolutional neural network; self-attention; deep learning

check for updates

**Citation:** Su, S.; Chen, R.; Fang, X.; Zhu, Y.; Zhang, T.; Xu, Z. A Novel Lightweight Grape Detection Method. *Agriculture* **2022**, *12*, 1364. https://doi.org/10.3390/agriculture12091364

Academic Editor: Dimitre Dimitrov

Received: 14 July 2022
Accepted: 31 August 2022
Published: 1 September 2022

## 1. Introduction

The grape has rich nutritional value and good taste, and it is widely popular among people. As an important part of the fruit industry, grape harvesting is labor-intensive and time-consuming [1]. Traditional manual picking is no longer sufficient to meet the needs of the fruit industry since the population is aging and the agricultural labor force is decreasing. It is urgent to develop automated grape-picking machines to harvest grapes in the field. Identifying and locating grapes in real-time is the first step to automating grape harvesting. However, traditional machine learning methods and deep learning-based grape detection methods fall short of practical requirements in speed and accuracy. Hence, developing a rapid and accurate grape detection method has great significance.

Numerous traditional methods have been proposed for grape detection in orchards in recent years. A grape image segmentation method [2] based on different color spaces was proposed and achieved a high recognition rate, but it did not consider the effect of leaf occlusion. Based on k-means clustering, Luo et al. [3] segmented stacked grapes to capture their contours and eventually achieved an 88.89% recognition rate. However, the recognition process required a great deal of time and could not meet the need for real-time grape detection. Pérez-Zavala et al. [4] used a support vector machine to classify information, combining shape and texture information, and achieved high precision and recall, but this method could not detect different varieties of grapes in complex scenes.

With the rapid development of deep learning, various convolutional neural network-based object detection methods have been applied to fruit detection tasks, with good detection results [5,6]. Some researchers investigated the effect of fruit datasets of scales and image resolutions on detection accuracy [7]. Parvathi et al. [8] took ResNet50 as the backbone network of Faster R-CNN and used it for coconut ripeness detection, achieving an accuracy of 89.4%. Fu et al. [9] proposed to remove the background from apple images by depth features before using Faster R-CNN for apple detection, which led to an average detection accuracy of 89.3%, and the detection time was 0.181 s. Faster R-CNN is a two-stage object detection method that has a high detection accuracy in fruit detection tasks [10], but the detection speed cannot meet the requirements. Some researchers proposed using one-stage object detection methods for fruit detection [11,12] to solve the problem. Aguiar et al. [13] used MobileNetV1 and Inception-V2 as networks for SSD and trained them using grape images at the different growth stages, achieving good detection results. Xiong et al. [14] added a residual network to the YOLOv3 model. They used the improved model for citrus recognition at night, resulting in a 2.27% improvement in average detection accuracy and a 26% increase in detection speed. Kateb et al. [15] proposed a modified attention mechanism based on the YOLO architecture to improve the stability of the detection model. In addition, they proposed a blackout regularization to provide better detection capability. Wu et al. [16] added a depth-separable convolution to the YOLOv3 model to improve the real-time detection performance of the model. Li et al. [17] improved the YOLOv4-tiny model by introducing the attention block and Soft-NMS algorithm into the network, increasing the identification accuracy. Meanwhile, the standard convolutions in the YOLOv4-tiny are replaced by depth-separable convolutions, improving the detection speed.

Compared with traditional methods for grape detection, the above detection methods usually utilize CNN as the backbone network to extract features and achieve good performance. However, the limited receptive field of convolutional kernels makes it hard to capture global dependency. Up to now, far too little attention has been paid to Vision Transformer. Vision Transformer models [18–20] take the self-attention to build a connection between pixels, obtaining the complete information of the image. Therefore, using Transformer as a backbone network is an intuitive way to enhance the detection performance. Additionally, feature fusion networks fuse high-level semantic information with low-level detailed information. Adding original features to the fused feature maps is an effective way to enrich the feature information. In post-processing, the bounding box with the highest confidence score is selected as the optimal bounding box. However, the low correlation between the confidence score and positioning accuracy [21] resulted in an optimal bounding box that could not surround the grape well. Thus, proposing a new Non-Maximum Suppression algorithm [22] is to be expected.

The objective of this study is to propose a novel lightweight grape detection method based on YOLOv4 [23] architecture. Inspired by the global information capture ability in Vision Transformer models, a hybrid convolution and transformer network called Uniformer [24] can be used as the backbone network of our method, which combines the advantages of convolution and self-attention to improve the feature extraction capability. To fuse more feature information, a novel feature fusion network based on the Path Aggregation Network (PANet) [25] is desired to be proposed. Compared with PANet, the feature fusion network has fewer parameters and computational complexity. Finally, we expect to propose a Relocation Non-Maximum Suppression (R-NMS) algorithm to improve the localization accuracy of the optimal bounding boxes.

## 2. Materials and Methods

### *2.1. Dataset*

The Wine Grape Instance Segmentation Dataset (WGISD) [26] was used as the experimental object. The dataset consists of images captured by different camera devices, and it has 5 grape varieties and 300 images, including 240 images with 2048 × 1365 resolution and 60 images with 2048 × 1536 resolution. Some grape images under different scenes are shown in Figure 1. In the experiment, these images were divided into a training set and a test set in the ratio of 4:1 for training and testing of the model: the training set had 240 images, and the test set had 60 images. A total of 4432 annotation boxes were annotated in the WGISD. The specific information of each grape annotation box is shown in Table 1. To further demonstrate the robustness of our method, another grape dataset called wGrapeUNIPD-DL [27] was used. There are 186 images with a resolution of 4288 × 2848, 17 images with a resolution of 4608 × 3456, and 65 images with a resolution of 4032 × 3024. The detailed information about the dataset is shown in Table 2. We divided this dataset in the ratio of 4:1 to obtain 214 images in the training set and 54 images in the test set. As shown in Figure 2, the grapes in this dataset have similar colors to the leaves under occlusion scenes, which brings some difficulty to the grape detection tasks.

**Table 1.** The detailed information about the WGISD.

| Categories | Species of Grapes | | | | | Total Number | Number of the Training Set | Number of the Test Set |
|---|---|---|---|---|---|---|---|---|
| | Chardonnay | Cabernet Franc | Cabernet Sauvignon | Sauvignon Blanc | Syrah | | | |
| Number of images | 65 | 65 | 57 | 65 | 48 | 300 | 240 | 60 |
| Number of labeled grapes | 840 | 1069 | 643 | 1317 | 563 | 4432 | 3500 | 932 |

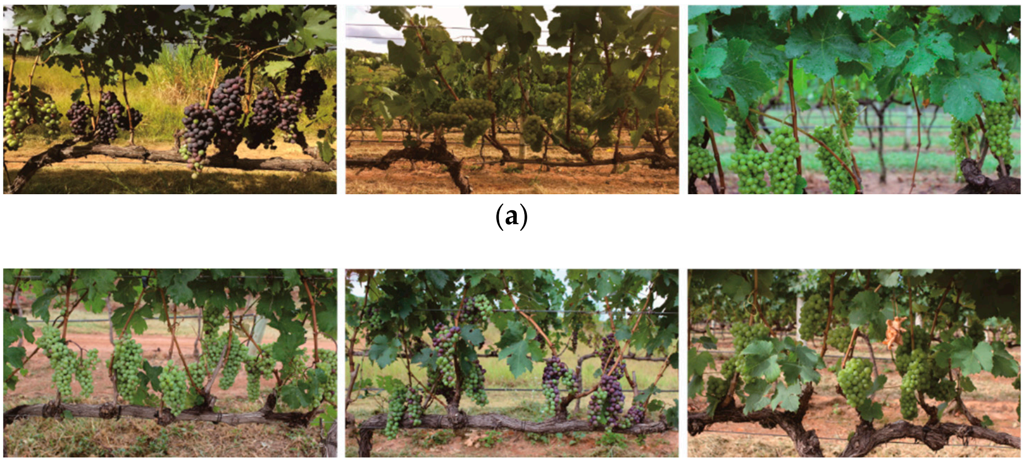


**Figure 1.** Grapes under different light, color, and overlapping scenes. (**a**) Grapes under sunny, cloudy, and rainy. (**b**) Green grapes, purple grapes, and overlapping grapes.

**Table 2.** The detailed information about the wGrapeUNIPD-DL.

| Categories | Total Number | Number of the Training Set | Number of the Test Set |
|---|---|---|---|
| Number of images | 268 | 214 | 54 |
| Number of labeled grapes | 2155 | 1744 | 411 |

**Figure 2.** Grapes under different light conditions.

### 2.2. Method

This study aims to design a novel lightweight grape detection method. As shown in Figure 3, our method has three parts. First, Uniformer was used as the backbone network to build the correlation between all pixels to enhance the feature extraction ability. Then, the BiPANet was proposed to fuse the different scale feature maps to enrich the feature information. For the last part, we proposed the R-NMS algorithm to enhance the localization accuracy of the optimal bounding boxes.

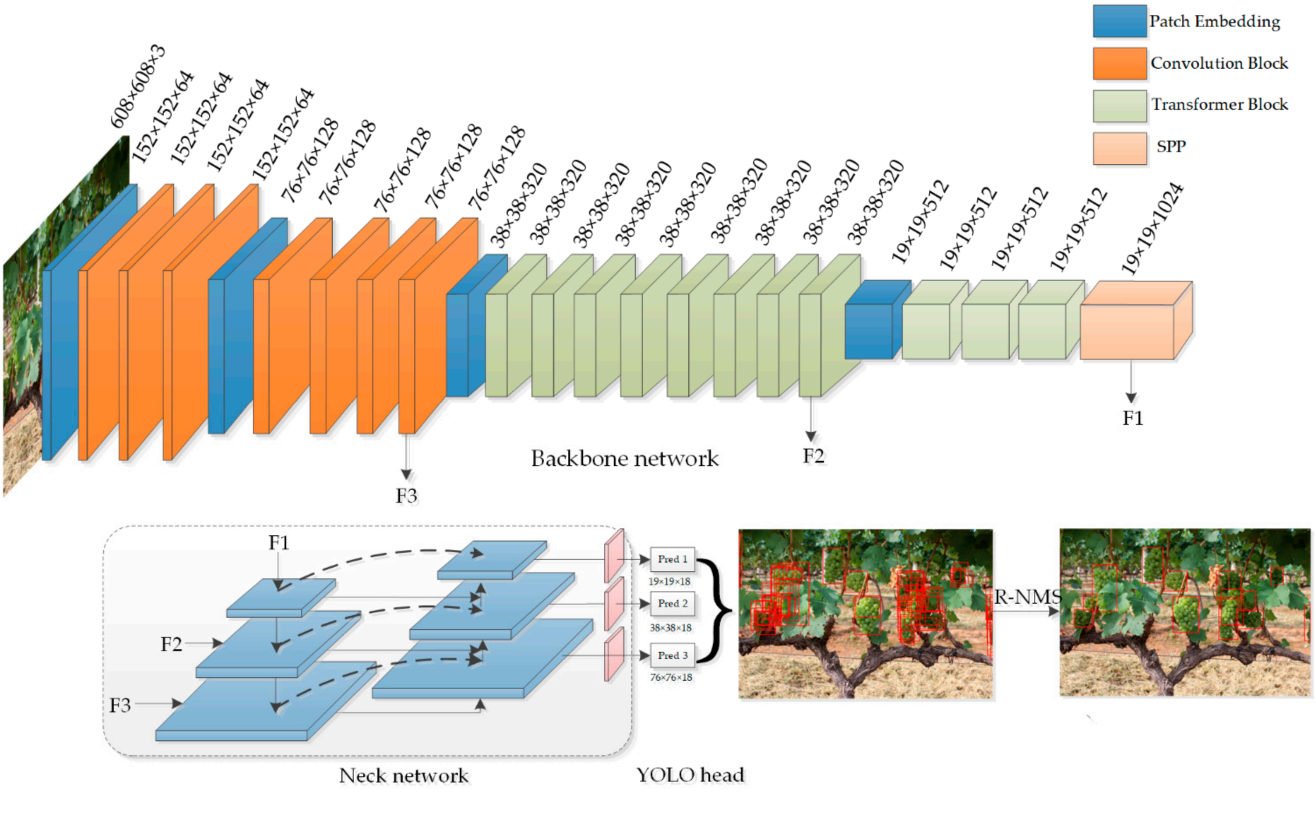


**Figure 3.** The network structure of our method.

#### 2.2.1. Backbone Network

Generally, CNN is used as the backbone network of various object detection methods, and it builds global perceptual fields by stacking multiple convolutional layers. However, stacking the convolutional layers tends to make the backbone network have a large number of parameters and high computational complexity. To make the backbone network model global information efficient with fewer parameters and low computational complexity, Uniformer was used as the backbone network of our method. As shown in Table 3, Uniformer contains four stages, and each stage includes one patch-embedding layer. In the first and second stages, there are three and four Convolution Blocks, respectively. In the last two stages, there are eight and three Transformer Blocks, respectively. The patch-embedding layer is used to compress the information in the spatial dimension to the channel dimension, reducing the parameters and computational complexity for the next step.

**Table 3.** Model configurations for the backbone network of our method.

| Stage | Input | Operation | Output |
|---|---|---|---|
| Stage 1 | 608 × 608 × 3 | $4 \times 4, 64, stride = 4$<br>$\begin{bmatrix} [3 \times 3, 64], \begin{bmatrix} 1 \times 1, 64 \\ 5 \times 5, 64 \\ 1 \times 1, 64 \end{bmatrix}, \begin{bmatrix} 1 \times 1, 256 \\ 1 \times 1, 64 \end{bmatrix} \end{bmatrix} \times 3$ | 152 × 152 × 64 |
| Stage 2 | 152 × 152 × 64 | $2 \times 2, 128, stride = 2$<br>$\begin{bmatrix} [3 \times 3, 128], \begin{bmatrix} 1 \times 1, 128 \\ 5 \times 5, 128 \\ 1 \times 1, 128 \end{bmatrix}, \begin{bmatrix} 1 \times 1, 512 \\ 1 \times 1, 128 \end{bmatrix} \end{bmatrix} \times 4$ | 76 × 76 × 128 |
| Stage 3 | 76 × 76 × 128 | $2 \times 2, 320, stride = 2$<br>$\begin{bmatrix} [3 \times 3, 320], [MHSA, 320], \begin{bmatrix} 320, 1280 \\ 1280, 320 \end{bmatrix} \end{bmatrix} \times 8$ | 38 × 38 × 320 |
| Stage 4 | 38 × 38 × 320 | $2 \times 2, 512, stride = 2$<br>$\begin{bmatrix} [3 \times 3, 512], [MHSA, 512], \begin{bmatrix} 512, 2048 \\ 2048, 512 \end{bmatrix} \end{bmatrix} \times 3$ | 19 × 19 × 512 |

The network structure of the Convolution Block is shown in Figure 4a. It is composed of the Dynamic Position Encoding (DPE) module, the Local Attention (LA) module, and the Feed Forward Network (FFN) module. To prevent network degradation, a residual skip connection was inserted in these layers. Convolution Block is calculated as follows:

$$\begin{aligned} X &= \mathrm{DPE}(X_{in}) + X_{in} \\ Y &= \mathrm{LA}(\mathrm{Norm}(X)) + X \\ Z &= \mathrm{FFN}(\mathrm{Norm}(Y)) + Y \end{aligned} \tag{1}$$

where $X_{in} \in \mathbb{R}^{C \times H \times W}$ represents the input feature map. The feature map passes through the DPE module to obtain the output feature map $X \in \mathbb{R}^{C \times H \times W}$. The DPE module was used to encode the position of the feature maps, whose convolution kernel size is 3 × 3. Then, the feature map $X \in \mathbb{R}^{C \times H \times W}$ was used as input to the LA module. To speed up the convergence of the model during training, the Norm layer was introduced into the LA module and the FFN module. We used the Norm layer to batch normalize the feature map $X \in \mathbb{R}^{C \times H \times W}$ and then input it to the LA module to obtain the feature map $Y \in \mathbb{R}^{C \times H \times W}$. The LA module was used to extract local features, and consists of two 1 × 1 Point-Wise Convolution (PWConv) layers and a 3 × 3 Depth-Wise Convolution (DWConv) layer. The structure of PWConv-DWConv-PWConv in the LA module comes from MobileNet [28]. Compared with standard convolution, it has fewer parameters. Finally, the feature map $Y \in \mathbb{R}^{C \times H \times W}$ was processed by the FFN module to obtain the final output feature map $Z \in \mathbb{R}^{C \times H \times W}$. The FFN module uses convolution to enhance the expression ability of features, and the convolution kernel sizes are both 1 × 1.

The Convolution Block uses the DPE module, LA module, and FFN module to extract the local information. However, it cannot capture long-range dependencies. To overcome this shortcoming, the Transformer Block was proposed. As shown in Figure 4b, the Transformer Block comprises the DPE module, the Global Attention (GA) module, and the Multi-Layer Perceptron (MLP) module. The Transformer Block is calculated as follows:

$$\begin{aligned} X &= \mathrm{DPE}(X_{in}) + X_{in} \\ Y &= \mathrm{GA}(\mathrm{Norm}(X)) + X \\ Z &= \mathrm{MLP}(\mathrm{Norm}(Y)) + Y \end{aligned} \tag{2}$$

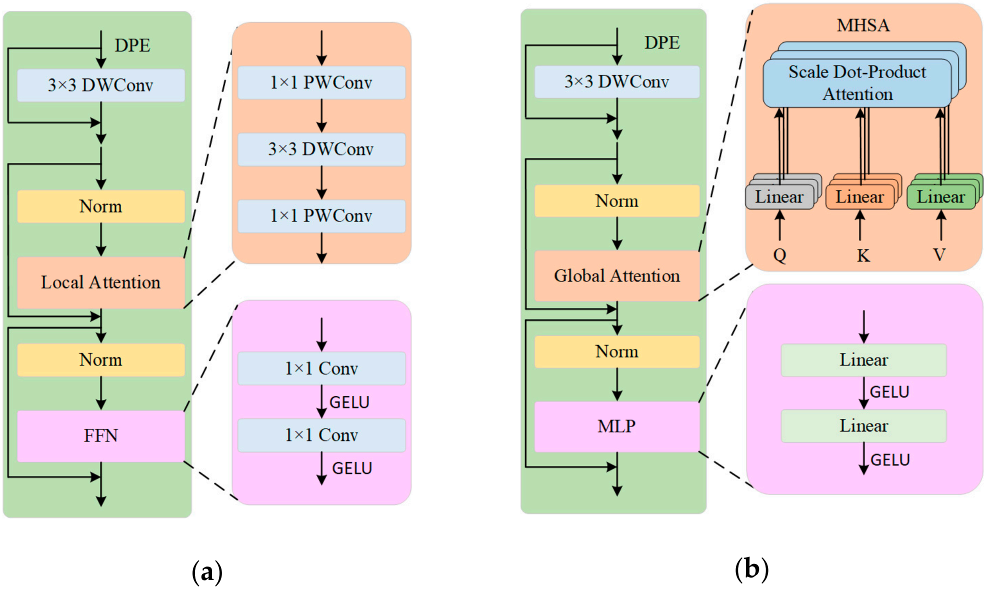


**Figure 4.** The network structure of the Convolution Block and the Transformer Block. (**a**) Convolution Block and (**b**) Transformer Block.

First, the DPE module encoded the position of the feature map $X_{in} \in \mathbb{R}^{C\times H\times W}$ using a 3 × 3 convolution kernel to obtain the feature map $X \in \mathbb{R}^{C\times H\times W}$. Then, the GA module and MLP module were used to process the feature maps $X \in \mathbb{R}^{C\times H\times W}$ and $Y \in \mathbb{R}^{C\times H\times W}$, respectively. Unlike the Convolution Block, the Transformer Block uses the GA module and MLP module to extract features and enhance them. The GA module uses self-attention to capture long-range information. Therefore, introducing the Transformer Block into the network can increase the feature extraction ability. The MLP module contains two full connection layers and two GELU functions, which can enhance the expression ability of features. Therefore, introducing the Transformer Block into the network can increase the ability to model global information.

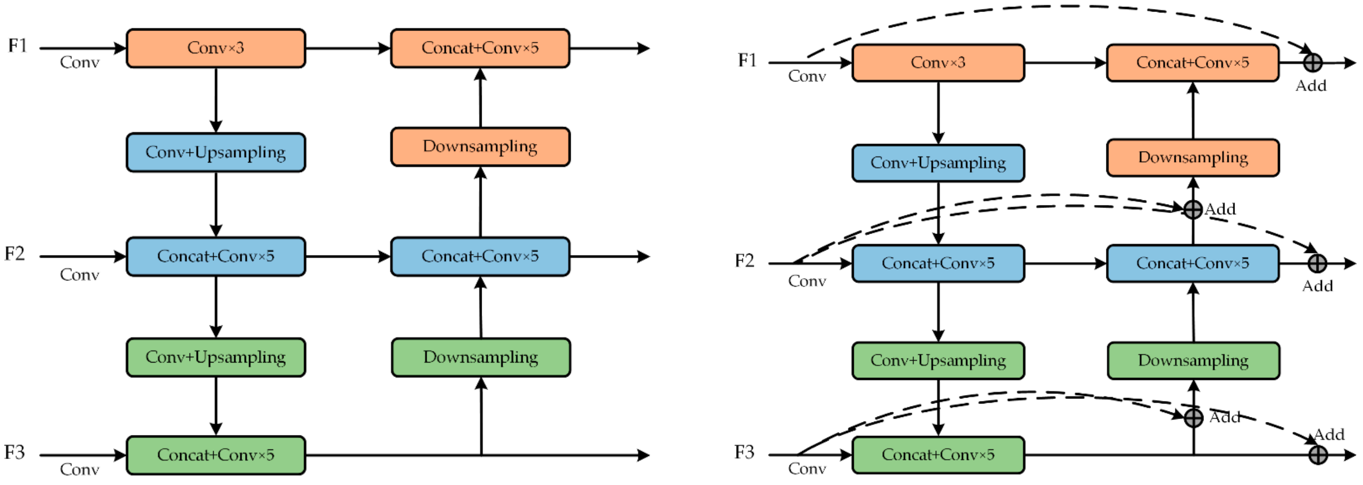
Compared with CNN, which needs to stack multiple convolutional layers to expand the perceptual field for global information, Uniformer can model global information with only a GA module. Ultimately, Uniformer can fully extract global feature information with fewer parameters and computational complexity and provides more useful feature information with the neck network.

### 2.2.2. Neck Network

As the backbone network of our method, Uniformer takes advantage of self-attention to sufficiently extract feature information. The low-level feature maps contain detailed information, and the high-level feature maps consist of semantic information. To enrich the feature information, PANet was proposed and used for multi-level feature fusion. The network structure of PANet is shown in Figure 5a. There are two pathways in PANet. To obtain more semantic information in low-level feature maps, PANet up-samples the low-resolution feature map in the top-down pathway. Then, the up-sampled feature map is concatenated with the feature map of the next layer. This process is iterated until the highest-resolution map fusion progress is finished. In the bottom-up pathway, PANet down-samples the high-resolution feature map and fuses it with the previous level feature map, enriching the detailed information in the high-level feature map. However, PANet ignores that the fused feature maps contain only a small amount of original feature information.

To fuse more feature information, the effective way is to add the original feature to the fused feature maps. Therefore, we proposed a cross-layer feature enhancement strategy and further constructed BiPANet based on PANet. The network structure of BiPANet is shown in Figure 5b. Compared with PANet, there are some cross-layer feature map fusion pathways in BiPANet. These pathways are used to add the original feature to the fused feature map. As a feature reuse approach, the cross-layer feature enhancement strategy can increase feature information in the feature map with almost no increase in computational cost. Consequently, BiPANet can fuse more information at different scale feature maps, and the fused feature maps contain richly detailed and semantic information.

The structure difference between PANet and BiPANet is described above. The specific construction process of BiPANet in our method is as follows. First, to reduce the number of parameters and computational complexity of the model, we obtained PANet-Lite by reducing the number of convolutional kernels in PANet. Compared with PANet, the smaller number of parameters and the computational complexity in PANet-Lite led to a small decrease in detection performance. Then, we further constructed BiPANet based on PANet-Lite. Reusing features can reduce the effect on network performance as the number of network parameters and computational complexity decrease [29]. Although the number of parameters and computational complexity of BiPANet were reduced compared with PANet, the feature reuse characteristic in the cross-layer feature enhancement strategy improved the feature expression, enhancing the detection performance. Ultimately, compared with PANet, BiPANet has better detection performance with fewer parameters and less computational complexity. To demonstrate the effectiveness of BiPANet, the ablation experiments were performed, as discussed in Section 3.3.



(**a**) (**b**)

**Figure 5.** The network structure of PANet and BiPANet. Conv denotes convolution. (**a**) PANet and (**b**) BiPANet. Up-sampling represents reducing the scales of the feature maps to twice their original scales. In contrast to up-sampling, down-sampling represents enlarging the scales of the feature maps to twice the original scales. Add means that the values in two feature maps are summed.

### 2.2.3. Bounding Box Prediction

The YOLO head uses multi-scale feature maps from BiPANet to predict the position of grapes, and a large number of bounding boxes were obtained. Figure 6 shows these bounding boxes with a significant amount of redundancy. In post-processing, the Non-Maximum Suppression (NMS) algorithm was used to preserve and suppress these bounding boxes. The algorithm selects the bounding box with the highest confidence score as the optimal bounding box. The low correlation between the confidence score and the localization accuracy of bounding boxes results in the optimal bounding box selected by the NMS algorithm, which cannot surround the grapes well.

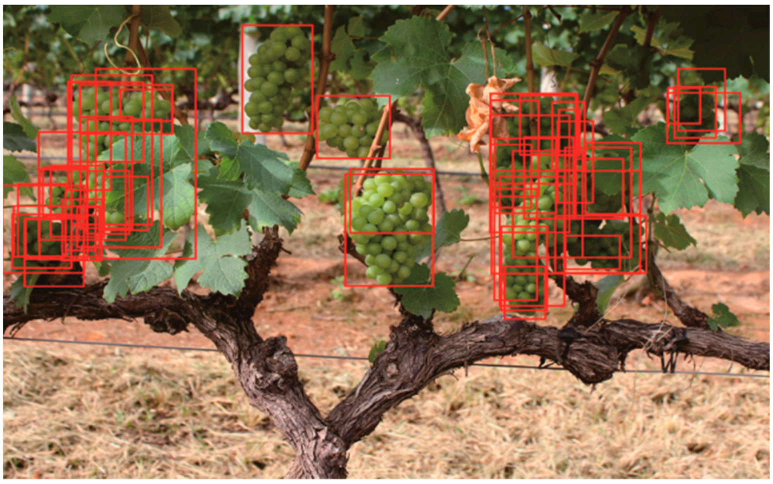


**Figure 6.** The predicted bounding boxes.

To improve the localization accuracy of the optimal bounding boxes, we proposed the R-NMS algorithm, which uses the location information of the bounding boxes around the optimal bounding boxes to perform the repositioning operations on the optimal bounding boxes. The new optimal bounding boxes have better localization accuracy than the former optimal bounding boxes. The Manhattan distance measures the proximity between two bounding boxes:

$$\begin{aligned} MH(u,v) &= |y_1 - q_1| + |x_1 - p_1| \\ MH(m,n) &= |y_2 - q_2| + |x_2 - p_2| \\ MH &= MH(u,v) + MH(m,n) \end{aligned} \tag{3}$$

Figure 7 shows the Manhattan distance between two bounding boxes. Since the Manhattan distance cannot accurately measure their overlap degree when their scales are significantly different, we normalized the bounding box coordinates. The process of coordinates' normalization is as follows:

$$\begin{aligned} X &= \{x_1, x_1, p_1, p_2\} \\ Y &= \{y_1, y_1, q_1, q_2\} \\ norm(x_i, y_i) &= \left(\frac{x_i - \min(X)}{\max(X) - \min(X)}, \frac{y_i - \min(Y)}{\max(Y) - \min(Y)}\right) \end{aligned} \tag{4}$$

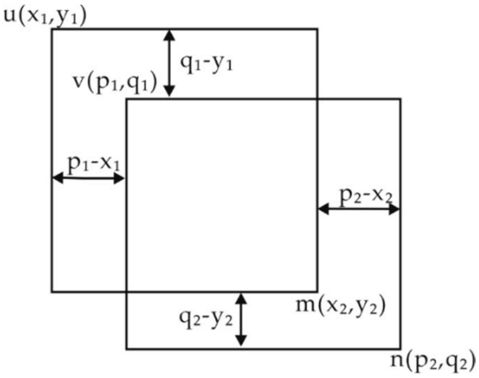


**Figure 7.** The Manhattan distance between two boxes. *u* and *m* are the upper-left and lower-right coordinates of box 1, respectively. *v* and *n* are the upper-left and lower-right coordinates of the box 2, respectively.

In Formula (4), the set of horizontal and vertical coordinates are denoted $X$ and $Y$, respectively, and the function *norm* is used to normalize the coordinates. After normalization, the Manhattan distance between two bounding boxes was computed. Then, we used the position information of the bounding boxes around the optimal bounding to relocate the optimal bounding box. The repositioning operation proceeds as follows:

$$\begin{gathered} MH(B_i, M) \leq P_{tr}, i \in (1, 2, \ldots, n) \\ O = \frac{\sum_{i=1}^{n} |B_i - M|}{n} \\ M_R = M + O \end{gathered} \quad (5)$$

In Formula (5), $M$ means the optimal bounding box, and $B_i, i \in (1, 2, \ldots, n)$ denotes the bounding boxes whose Manhattan distance from the optimal bounding box is lower than the threshold $P_{tr}$. $O$ represents the average offset between those bounding boxes around the optimal bounding box. The optimal bounding box $M$ is repositioned with the offset to obtain the new optimal bounding box $M_R$. The experimental results of the R-NMS algorithm on the WGISD are analyzed in Section 3.6. to show how well it performed on the grape detecting task.

## 3. Results and Discussion

### 3.1. Implementation Details

In our experiment, we used GPU GeForce RTX 2080Ti to accelerate model training, and the CPU was AMD Ryzen 9 3900X. The operating system was Ubuntu18.04, and the programming language was Python 3.8. The other six models in the experiments were implemented using Pytorch, and their code was obtained from GitHub (Code available at: https://github.com/bubbliiiing/ (accessed on 31 May 2022)). The main code of our model was derived from the YOLOv4 model, and the code and pre-trained model of Uniformer were obtained from the Uniformer research team (code and pre-trained model available at: https://github.com/SenseX/UniFormer (accessed on 31 May 2022)). All models used the same training parameters and were trained by pre-trained models during training, and none of them used image enhancement strategies.

The various parameters set during the model training are shown in Table 4. We used Adam gradient descent to train models. A total of 150 training epochs were set, and we divided them into two phases. The training strategy of freezing the parameters in the backbone network was adopted for the first 75 epochs. In this phase, the learning rate was set to 0.001, the weight decay rate was set to 0.0005, and the batch size was set to 8. In the last 75 epochs, the learning rate was set to 0.0001, the weight decay rate was set to 0.0005, and the batch size was set to 4. In the model test phase, IOU and classification confidence were set to 0.5 and 0.001, respectively, and the threshold, $P_{tr}$, in the R-NMS algorithm was set to 0.6.

**Table 4.** The setting of training model parameters.

| Parameters | Values |
|---|---|
| Image resolution | 608 × 608 |
| Batch size 1 | 8 |
| Batch size 2 | 4 |
| Learning rate 1 | 0.001 |
| Learning rate 2 | 0.0001 |
| Weight decay rate 1 | 0.0005 |
| Weight decay rate 2 | 0.0005 |
| Optimizer | Adam |
| Epochs | 150 |

### 3.2. Evaluation Metrics

We used seven evaluation metrics: precision, recall, F1 score, mean Average Precision (mAP), Frames Per Second (FPS), Params, and Floating-Point Operations (FLOPs), to evaluate the performance of our method and other popular grape detection methods.

The recall represents the proportion of positive samples with correct predictions to all positive samples and is calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

*TP* indicates that positive samples were predicted as positive samples, and *FN* indicates that positive samples were predicted as negative samples. Precision represents the proportion of true positive samples among all predicted positive samples and is calculated as shown in Equation (7).

*FP* indicates predicting negative samples as negative samples. The average precision (*AP*) is used as a composite measure of the model performance. The equation of *AP* is as follows:

$$AP = \int_0^1 P(R)dR \tag{8}$$

$$mAP = \frac{\sum_{i=1}^{n} AP_i}{n} \tag{9}$$

there is only one class: the grape class, in the grape datasets, so *mAP* = *AP*.

$F1$ score is used as an evaluation metric for the combined measure of accuracy and recall, with the following equation:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{10}$$

Params mean the number of parameters in the algorithm. Algorithm complexity is measured by GFLOPs. FPS represents the number of images detected by the algorithm per second.

### 3.3. Ablation Experiments

The effectiveness of Uniformer, BiPANet, and the R-NMS is examined in this section through a series of ablation experiments. Tables 5 and 6 present the experimental results of various methods on the WGISD and wGrapeUNIPD-DL. A represents the baseline YOLOv4, and B was obtained by replacing the backbone network CSPDarknet53 with Uniformer. In post-processing, the NMS algorithm was used in A and B. The evaluation metrics of B on the WGISD and wGrapeUNIPD-DL were better than A. Uniformer takes advantage of Convolution and Transformer to improve the feature extraction ability of the network. As a result, the performance of B was improved. In addition, compared to CSPDarknet53, Uniformer has fewer parameters and low computation complexity, which made the FPS of B higher than that of A. C was improved based on B. C uses PANet-Lite to replace the PANet in B. PANet-Lite was obtained by reducing the number of convolution kernels in PANet. Compared with PANet, PANet-Lite has a smaller number of parameters and lower computational complexity, resulting in the performance of C being poorer than B on the WGISD and wGrapeUNIPD-DL. To solve the problem, we proposed a cross-layer feature enhancement strategy and further constructed BiPANet. D was constructed by replacing PANet-Lite in C with BiPANet. Attributed to the cross-layer feature enhancement strategy, more feature information was fused by BiPANet, so the mAP of D was better than B and C on the WGISD and wGrapeUNIPD-DL. Compared with B, E used the R-NMS algorithm to retain and suppress the candidate bounding boxes in post-processing and achieved good performance. The algorithm used the position information of the bounding boxes around the optimal bounding boxes and performed the repositioning operations on the optimal bounding boxes to improve the localization accuracy. However, the process is time-consuming and leads to decreased FPS. Unlike D, our method used the R-NMS algorithm to retain and suppress the candidate bounding boxes. The R-NMS algorithm improved the localization accuracy of the optimal bounding boxes and reduced the redundant bounding boxes, so the mAP, precision, and recall of our method were increased. Compared with the NMS algorithm, the R-NMS algorithm increased the mAP, and the FPS decreased from 50 to 46. The 46 FPS can also meet the real-time detection requirement. For detection tasks with high real-time requirements, we can utilize tensorRT with the R-NMS algorithm implemented by using GPU to speed up the inference. Our method makes it possible to have high detection accuracy with a fast detection speed.

**Table 5.** Experimental results of different methods on the WGISD.

| Methods | Uniformer | PANet-Lite | BiPANet | R-NMS | Precision | Recall | F1 | mAP | Params | FLOPs | FPS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| A | ✗ | ✗ | ✗ | ✗ | 87.2% | 76.5% | 81.5% | 86.0% | 64.0 M | 63.9 G | 38 |
| B | ✓ | ✗ | ✗ | ✗ | 87.4% | 77.5% | 82.2% | 87.0% | 54.0 M | 51.2 G | 44 |
| C | ✓ | ✓ | ✗ | ✗ | 87.2% | 77.2% | 81.9% | 86.7% | 34.8 M | 36.3 G | 50 |
| D | ✓ | ✗ | ✓ | ✗ | 87.7% | 77.7% | 82.4% | 87.3% | 34.8 M | 36.3 G | 50 |
| E | ✓ | ✗ | ✗ | ✓ | 87.9% | 78.1% | 83.1% | 87.5% | 54.0 M | 51.2 G | 39 |
| Our method | ✓ | ✗ | ✓ | ✓ | 88.6% | 78.3% | 83.1% | 87.7% | 34.8 M | 36.3 G | 46 |

**Table 6.** Experimental results of different methods on the wGrapeUNIPD-DL.

| Methods | Uniformer | PANet-Lite | BiPANet | R-NMS | Precision | Recall | F1 | mAP | Params | FLOPs | FPS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| A | ✗ | ✗ | ✗ | ✗ | 84.3% | 60.6% | 70.5% | 70.4% | 64.0 M | 63.9 G | 38 |
| B | ✓ | ✗ | ✗ | ✗ | 85.0% | 61.8% | 71.6% | 72.0% | 54.0 M | 51.2 G | 44 |
| C | ✓ | ✓ | ✗ | ✗ | 85.5% | 60.4% | 70.8% | 71.7% | 34.8 M | 36.3 G | 50 |
| D | ✓ | ✗ | ✓ | ✗ | 85.2% | 62.4% | 72.0% | 72.4% | 34.8 M | 36.3 G | 50 |
| E | ✓ | ✗ | ✗ | ✓ | 85.4% | 62.0% | 71.8% | 72.2% | 54.0 M | 51.2 G | 39 |
| Our method | ✓ | ✗ | ✓ | ✓ | 85.7% | 62.3% | 72.2% | 72.8% | 34.8 M | 36.3 G | 46 |

### *3.4. Experimental Analysis of PANet and BiPANet*

In this section, we further compare and analyze the performance of PANet and BiPANet. In Tables 5 and 6, B, C, and D represent the models which use PANet, PANet-Lite, and BiPANet as the feature fusion network, with Uniformer as the backbone network, respectively. PANet-Lite was obtained by reducing the number of convolutional kernels in PANet. Compared with PANet, PANet-Lite has fewer parameters and less computational complexity. Compared with B, the number of parameters and the computational complexity of C were reduced by 19.2 M and 14.9 GFLOPs, respectively. The mAP of B and C on the WGISD dataset were 87.0%, and 86.7%, respectively. Compared with B, the mAP of C on the wGrapeUNIPD-DL was reduced by 0.3%. Experiments on the WGISD and wGrapeUNIPD-DL showed that C had a poorer detection performance than B. To overcome this shortcoming, we proposed a cross-layer feature enhancement strategy and further constructed BiPANet based on PANet-Lite. As a feature reuse approach, the cross-layer feature enhancement strategy can improve the representational power of the network with almost no increase in the number of parameters and computational complexity. Therefore, the detection performance of D was increased. Compared with B and C, the mAP of D on the WGISD was increased by 0.3% and 0.6%, respectively. The mAP of D on the wGrapeUNIPD-DL was 72.4%, which is higher than B and C. Besides, the FPS of D was 50, and the FPS of B was 44. Consequently, selecting BiPANet as the feature fusion network can increase the detection accuracy and the speed.

### *3.5. Experimental Results and Analysis*

In this section, we compare the performance of our method and some popular object detection methods on the WGISD and wGrapeUNIPD-DL. The experimental results are shown in Tables 7 and 8. Our method achieved an 87.7% mAP. Compared with Faster R-CNN, SSD, RetinaNet, YOLOv3, YOLOv4, and YOLOx, the mAP of our method was increased by 21.4%, 2.5%, 13.3%, 3.5%, 1.7%, and 0.8%, respectively. The precision was 72.8%, 85.1%, 78.9%, 83.4%, 87.2%, and 85.6% in Faster R-CNN, SSD, RetinaNet, YOLOv3, YOLOv4, and YOLOx, respectively, which are lower than our method. The number of parameters in our method, Faster R-CNN, SSD, RetinaNet, YOLOv3, YOLOv4, and YOLOx was 34.8 M, 28.3 M, 24.4 M, 36.5, 61.6 M, 64.0 M, and 54.2 M, respectively. On the wGrapeUNIPD-DL, the mAP of our method, Faster R-CNN, SSD, RetinaNet, YOLOv3, YOLOv4, and YOLOx was 72.8%, 43.2%, 56.2%, 45.7%, 65.6%, 70.4%, and 72.6%, respectively. The number of parameters in our method was not the least among all methods, but the precision, recall, F1, and mAP of our method outperformed the other methods. Besides, among all the compared methods, the computational complexity of our method was the smallest, only 36.3 G FLOPs, and the FPS of our method was one of the highest.

The detection results obtained by our method on the WGISD and wGrapeUNIPD-DL are shown in Figures 8 and 9, and our method could accurately identify and locate most grapes. Table 9 shows the detailed detection results of various methods for the images in Figures 8 and 9. There are 75 ground-truth bounding boxes in Figure 8, and our method detected 73 TP bounding boxes. The number of TP bounding boxes detected by Faster R-CNN, SSD, RetinaNet, YOLOv3, YOLOv4, and YOLOx was 68, 69, 66, 68, 70, and 72, respectively. The TN and FN bounding boxes detected by our method were 1 and 3, which is better than other methods. There are 21 ground-truth bounding boxes in Figure 9. Our method and YOLOx detected 39 ground-truth bounding boxes, and no TN and FN bounding boxes were found in the detection results. However, there were FN and PN bounding boxes in the detection results of the methods such as Faster R-CNN. The detection results in Figures 8 and 9 demonstrate that our method achieved good detection performance.

In our method, Uniformer was used as the backbone network. The backbone networks of the other methods are based on CNN, and the poor ability of CNN to model global information prevents them from fully extracting feature information. From the ablation experiment results in Section 3.3, when the backbone network was Uniformer, the mAP was increased by 1%, and the number of parameters and computational complexity were decreased by 10 M and 12.7 GFLOPs, respectively. Uniformer can capture long-range information to improve the feature extraction capability, which led to a significantly good performance in the detection accuracy of our method on the grape detection task. According to the experimental results in Table 5, due to the small number of convolutional kernels in BiPANet and the cross-layer feature enhancement strategy, BiPANet used with Uniformer improved the mAP from 87.0% to 87.3%, and the number of parameters and computational complexity were reduced by 19.2 M and 14.9 GLOPs, respectively. Additionally, the R-NMS algorithm used the position information of the bounding boxes around the optimal bounding boxes to perform repositioning operations on the optimal bounding boxes, which led to an improvement in the localization accuracy of the optimal bounding boxes. As a result, the precision, recall, and mAP of our method were increased. According to the above results, it can be concluded that our method is efficient and lightweight.

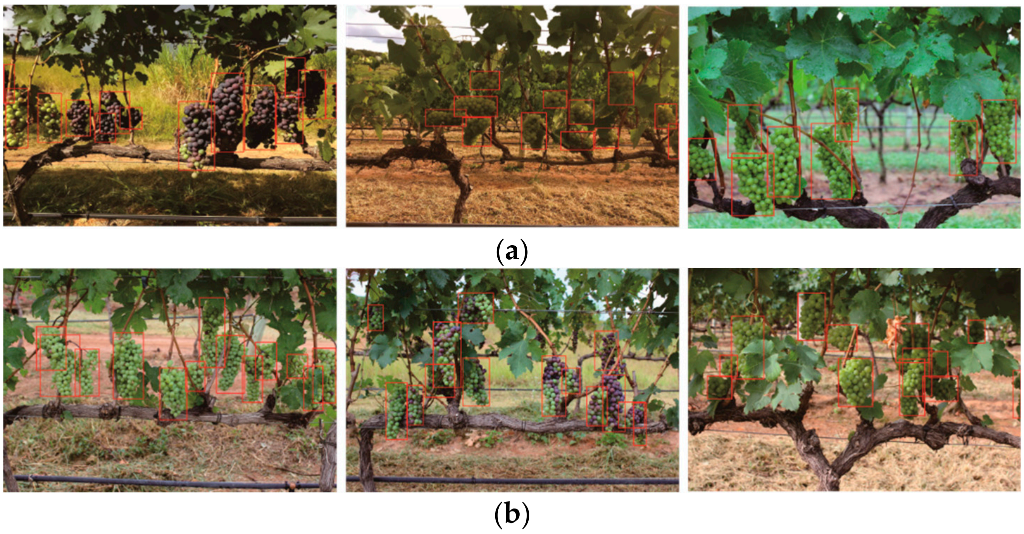


**Figure 8.** The detection results of our method on the WGISD. (**a**) NMS; (**b**) R-NMS.

**Table 7.** The experimental results on the WGISD using different methods.

| Methods | Precision | Recall | F1 | mAP | Params | FLOPs | FPS |
|---|---|---|---|---|---|---|---|
| Faster R-CNN | 72.8% | 51.6% | 60.4% | 66.3% | 28.3 M | 196.5 G | 20 |
| SSD | 85.1% | 75.9% | 80.2% | 85.2% | 24.4 M | 124.6 G | 46 |
| RetinaNet | 78.9% | 63.8% | 70.1% | 74.4% | 36.5 M | 74.7 G | 36 |
| YOLOv3 | 83.4% | 77.7% | 80.4% | 84.2% | 61.6 M | 70.0 G | 44 |
| YOLOv4 | 87.2% | 76.5% | 81.5% | 86.0% | 64.0 M | 63.9 G | 38 |
| YOLOx | 85.6% | 80.4% | 82.9% | 86.9% | 54.2 M | 70.1 G | 44 |
| Our method | 88.6% | 78.3% | 83.1% | 87.7% | 34.8 M | 36.3 G | 46 |

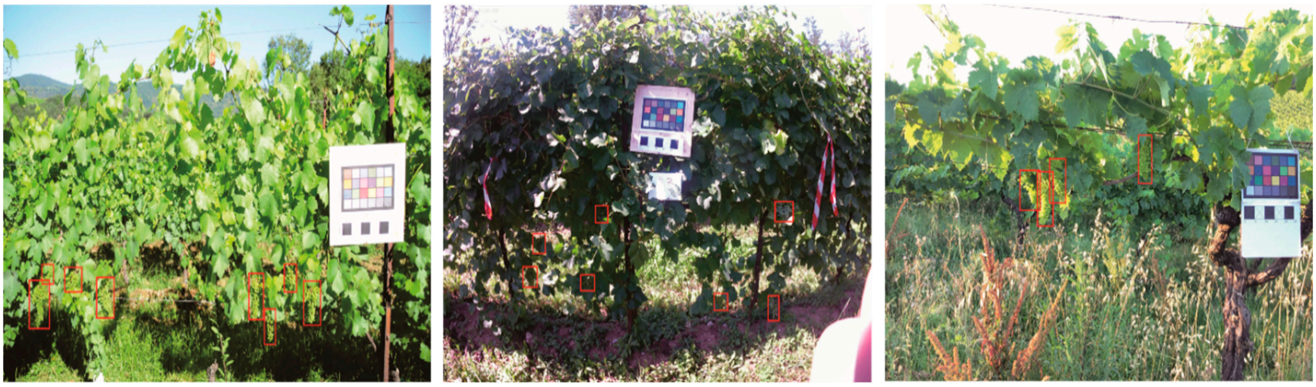


**Figure 9.** The detection results of our method on the wGrapeUNIPD-DL.

**Table 8.** The experimental results on the wGrapeUNIPD-DL using different methods.

| Methods | Precision | Recall | F1 | mAP | Params | FLOPs | FPS |
|---|---|---|---|---|---|---|---|
| Faster R-CNN | 65.7% | 40.2% | 49.9% | 43.2% | 28.3 M | 196.5 G | 20 |
| SSD | 72.8% | 51.9% | 60.6% | 56.2% | 24.4 M | 124.6 G | 46 |
| RetinaNet | 67.7% | 46.1% | 54.9% | 45.7% | 36.5 M | 74.7 G | 36 |
| YOLOv3 | 83.0% | 53.6% | 65.1% | 65.6% | 61.6 M | 70.0 G | 44 |
| YOLOv4 | 84.3% | 60.6% | 70.5% | 70.4% | 64.0 M | 63.9 G | 38 |
| YOLOx | 79.6% | 65.4% | 71.8% | 72.6% | 54.2 M | 70.1 G | 44 |
| Our method | 85.7% | 62.3% | 72.2% | 72.8% | 34.8 M | 36.3 G | 46 |

**Table 9.** The detection results on the WGISD and wGrapeUNIPD-DL using different methods.

| **Methods** | **Grape Detection Results in Figure 8** | | | **Grape Detection Results in Figure 9** | | |
|---|---|---|---|---|---|---|
| | **Number of TP Bounding Boxes** | **Number of FP Bounding Boxes** | **Number of FN Bounding Boxes** | **Number of TP Bounding Boxes** | **Number of FP Bounding Boxes** | **Number of FN Bounding Boxes** |
| Faster R-CNN | 68 | 22 | 41 | 15 | 11 | 10 |
| SSD | 69 | 6 | 10 | 12 | 0 | 0 |
| RetinaNet | 66 | 17 | 26 | 14 | 1 | 0 |
| YOLOv3 | 68 | 8 | 7 | 17 | 0 | 0 |
| YOLOv4 | 70 | 3 | 5 | 19 | 1 | 2 |
| YOLOx | 72 | 5 | 3 | 19 | 0 | 0 |
| Our method | 73 | 1 | 3 | 19 | 0 | 0 |

### 3.6. Experimental Analysis of the R-NMS Algorithm

This section analyzes the detection results under the R-NMS algorithm on the WGISD. The performance of the R-NMS algorithm depends on the setting of the threshold, $P_{tr}$. The experimental results under the R-NMS algorithm with different thresholds, $P_{tr}$, on the WGISD are shown in Figure 10. When the threshold, $P_{tr}$, was 0, the R-NMS algorithm degenerated to the NMS algorithm. As the threshold value increased, more valid position information of surrounding bounding boxes was obtained and used for the optimal bounding boxes' repositioning, improving the localization accuracy. The precision, recall, and mAP of our method were improved in this range. When the threshold, $P_{tr}$, was greater than 0.6, the number of bounding boxes surrounding the optimal bounding box increased. However, the added bounding boxes were far away from the optimal bounding box, and their position information was invalid and had a suppressing effect on the positioning accuracy. Consequently, the localization accuracy of the optimal bounding boxes by performing repositioning operations using this location information was reduced, which led to a decrease in the precision, recall, and mAP. To achieve a better detection performance, when we use the R-NMS algorithm in post-processing, the threshold, $P_{tr}$, should be set to about 0.5 if possible.

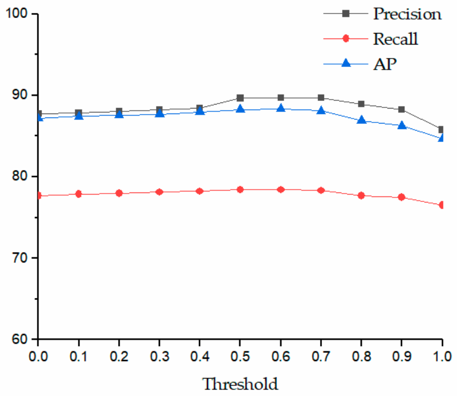


**Figure 10.** Detection accuracy under different thresholds,$P_{tr}$.

To obtain good performance, the threshold, $P_{tr}$, was set to 0.6. Compared with the NMS algorithm, the optimal bounding box obtained by the R-NMS algorithm had a higher localization accuracy and enclosed the grapes well. The result in Figure 11a was obtained by the NMS algorithm, and the result in Figure 11b was obtained by the R-NMS algorithm. The localization accuracy of the bounding box in Figure 11a was significantly better than that in Figure 11b. Figure 12 shows the detection results under the NMS and R-NMS algorithms. There was a redundant bounding box in Figure 12a. In the bounding box suppression phase, the IOU between the redundant bounding box and the optimal bounding box was less than the threshold, $P_{tr}$. Therefore, the redundant bounding box was not suppressed by the optimal bounding box under the NMS algorithm. In Figure 12b, the R-NMS algorithm performed a repositioning operation on the optimal bounding box by using the position information around the optimal bounding box and obtaining a new optimal bounding box. Since the IOU between the new optimal bounding box and the redundant bounding box was greater than the threshold, $P_{tr}$, the redundant bounding box was suppressed. Consequently, the R-NMS algorithm can improve the localization accuracy of the optimal bounding boxes and reduce redundant bounding boxes.

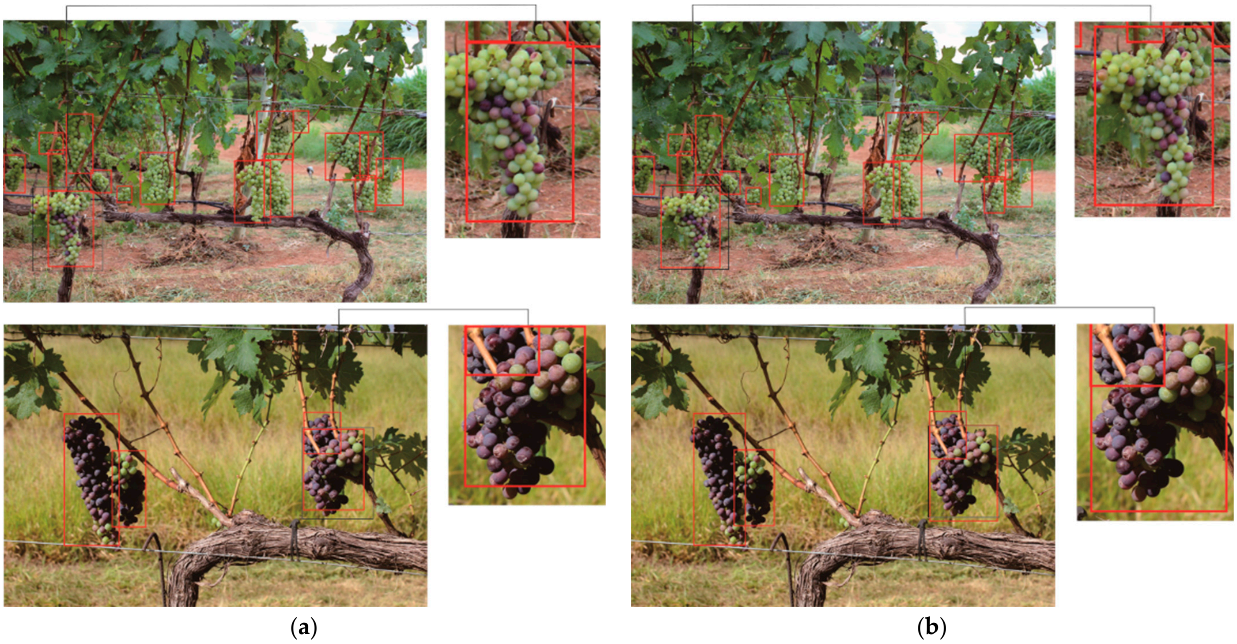


**Figure 11.** The detection results under different non-maximum suppression algorithms. (**a**) NMS; (**b**) R-NMS.

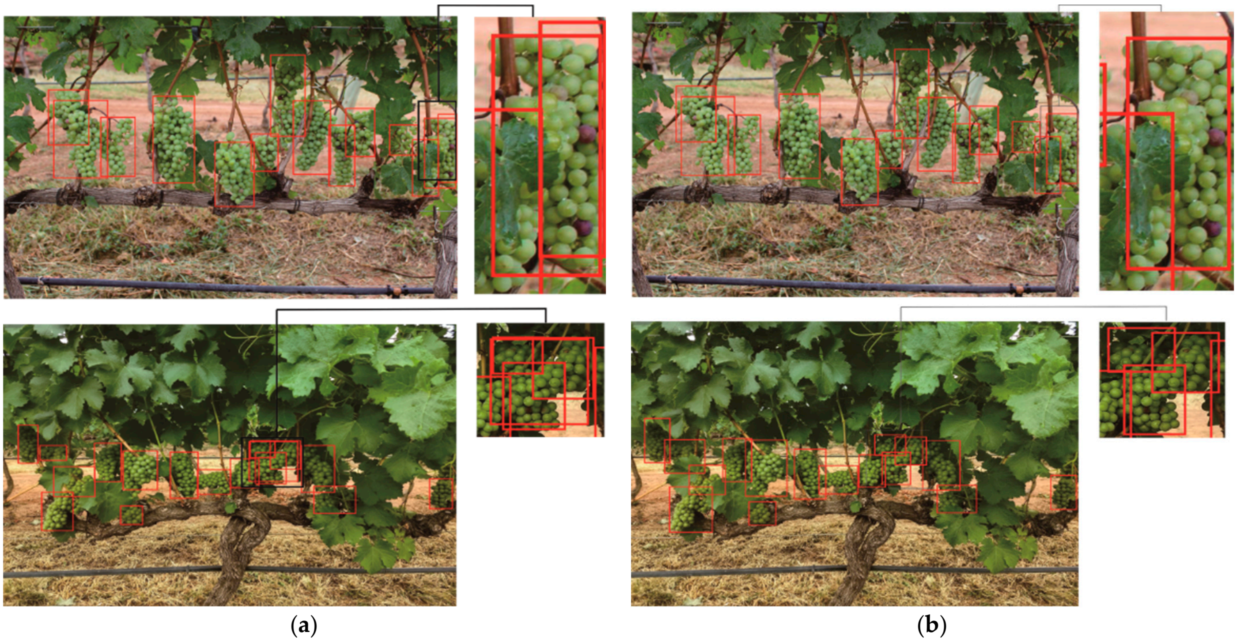


**Figure 12.** The detection results under different non-maximum suppression algorithms. (**a**) NMS; (**b**) R-NMS.

## 4. Conclusions

In this study, a novel lightweight grape detection method was proposed. The backbone network of our method is a hybrid Convolution and Transformer network called Uniformer, which builds global correlations between all pixels. Compared with CNN, Uniformer fully used the advantages of CNN and Transformer to improve the feature extraction capability with a small number of parameters and less computational complexity. Then, to embed more feature information in fused feature maps, the method provided a novel cross-layer feature enhancement strategy and further constructed BiPANet based on PANet. Due to the low correlation between localization accuracy and confidence, the optimal bounding boxes selected by NMS had low localization accuracy. To solve this problem, the method proposed the R-NMS algorithm. The algorithm used the position information of the bounding boxes around the optimal bounding boxes to perform the repositioning operations on the optimal bounding boxes, obtaining new optimal bounding boxes with high localization accuracy. Special ablation experiments were designed to verify the effectiveness of Uniformer, BiPANet, and the R-NMS algorithm in our method. The experiment results showed that all the algorithms can improve the detection performance. Besides, we analyzed the effect of the R-NMS algorithm from the extensive experimental results. According to those experiments, the R-NMS algorithm can improve the localization accuracy of the bounding boxes. Experimental results on the WGISD demonstrated that our method achieved an 87.7% mAP and 46 FPS, indicating that our method is an effective grape detection method with good accuracy and a fast detection speed.

**Author Contributions:** Funding acquisition, X.F.; Methodology, R.C.; Software, R.C. and Z.X.; Validation, Y.Z.; Writing—original draft, R.C.; Writing—review & editing, S.S. and T.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Institute of Energy, Hefei Comprehensive National Science Center (under Grant No. 19KZS203), the National Natural Science Foundation of China (No. 61806006), and the China Postdoctoral Science Foundation (No. 2019M660149).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Publicly available datasets were analyzed in this study. The data can be found here: https://github.com/thsant/wgisd, accessed on 20 April 2022.

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

## References

1. Peng, Y.; Wang, A.; Liu, J.; Faheem, M. A comparative study of semantic segmentation models for identification of grape with different varieties. *Agriculture* **2021**, *11*, 997. [CrossRef]
2. Ma, B.; Jia, Y.; Mei, W.; Gao, G.; Lv, C.; Zhou, Q. Study on the recognition method of grape in different natural environment. *Mod. Food Sci. Technol.* **2015**, *31*, 145–149. [CrossRef]
3. Luo, L.; Zou, X.; Wang, C.; Chen, X.; Yang, Z.; Situ, W. Recognition method for two overlapping and adjacent grape clusters based on image contour analysis. *Trans. Chin. Soc. Agric. Mach.* **2017**, *48*, 15–22. [CrossRef]
4. Pérez-Zavala, R.; Torres-Torriti, M.; Cheein, F.A.; Troni, G. A pattern recognition strategy for visual grape bunch detection in vineyards. *Comput. Electron. Agric.* **2018**, *151*, 136–149. [CrossRef]
5. Yan, B.; Fan, P.; Lei, X.; Liu, Z.; Yang, F. A real-time apple targets detection method for picking robot based on improved YOLOv5. *Remote Sens.* **2021**, *13*, 1619. [CrossRef]
6. Liu, F.; Liu, Y.; Lin, S.; Guo, W.; Xu, F.; Zhang, B. Fast recognition method for tomatoes under complex environments based on improved YOLO. *Trans. Chin. Soc. Agric. Mach.* **2020**, *51*, 229–237. [CrossRef]
7. Wang, X.; Tang, J.; Whitty, M. Data-centric analysis of on-tree fruit detection: Experiments with deep learning. *Comput. Electron. Agric.* **2022**, *194*, 106748. [CrossRef]
8. Parvathi, S.; Selvi, S.T. Detection of maturity stages of coconuts in complex background using Faster R-CNN model. *Biosyst. Eng.* **2021**, *202*, 119–132. [CrossRef]
9. Fu, L.; Majeed, Y.; Zhang, X.; Karkee, M.; Zhang, Q. Faster R–CNN–based apple detection in dense-foliage fruiting-wall trees using RGB and depth features for robotic harvesting. *Biosyst. Eng.* **2020**, *197*, 245–256. [CrossRef]

10. Gao, F.; Fu, L.; Zhang, X.; Majeed, Y.; Li, R.; Karkee, M.; Zhang, Q. Multi-class fruit-on-plant detection for apple in SNAP system using Faster R-CNN. *Comput. Electron. Agric.* **2020**, *176*, 105634. [CrossRef]
11. Peng, H.; Huang, B.; Shao, Y.; Li, Z.; Zhang, C.; Chen, Y.; Xiong, J. General improved SSD model for picking object recognition of multiple fruits in natural environment. *Trans. Chin. Soc. Agric. Eng.* **2018**, *34*, 155–162. [CrossRef]
12. Zhao, D.; Wu, R.; Liu, X.; Zhao, Y. Apple positioning based on YOLO deep convolutional neural network for picking robot in complex background. *Trans. Chin. Soc. Agric. Eng.* **2019**, *35*, 172–181. [CrossRef]
13. Aguiar, A.S.; Magalhães, S.A.; Dos Santos, F.N.; Castro, L.; Pinho, T.; Valente, J.; Martins, R.; Boaventura-Cunha, J. Grape bunch detection at different growth stages using deep learning quantized models. *Agronomy* **2021**, *11*, 1890. [CrossRef]
14. Xiong, J.; Zheng, Z.; Liang, J.E.; Zhong, Z.; Liu, B.; Sun, B. Citrus detection method in night environment based on improved YOLO v3 Network. *Trans. Chin. Soc. Agric. Mach.* **2020**, *51*, 199–206. [CrossRef]
15. Kateb, F.A.; Monowar, M.M.; Hamid, A.; Ohi, A.Q.; Mridha, M.F. FruitDet: Attentive feature aggregation for real-time fruit detection in orchards. *Agronomy* **2021**, *11*, 2440. [CrossRef]
16. Wu, X.; Qi, Z.; Wang, L.; Yang, J.; Xia, X. Apple detection method based on light-YOLOv3 convolutional neural network. *Trans. Chin. Soc. Agric. Mach.* **2020**, *51*, 17–25. [CrossRef]
17. Li, H.; Li, C.; Li, G.; Chen, L. A real-time table grape detection method based on improved YOLOv4-tiny network in complex background. *Biosyst. Eng.* **2021**, *212*, 347–359. [CrossRef]
18. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 213–229.
19. Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 568–578.
20. Wu, H.; Xiao, B.; Codella, N.; Liu, M.; Dai, X.; Yuan, L.; Zhang, L. Cvt: Introducing convolutions to vision transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 22–31.
21. Jiang, B.; Luo, R.; Mao, J.; Xiao, T.; Jiang, Y. Acquisition of localization confidence for accurate object detection. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 784–799.
22. Neubeck, A.; Van Gool, L. Efficient non-maximum suppression. In Proceedings of the 18th International Conference on Pattern Recognition, Hong Kong, China, 20–24 August 2006; pp. 850–855.
23. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
24. Li, K.; Wang, Y.; Zhang, J.; Gao, P.; Song, G.; Liu, Y.; Li, H.; Qiao, Y. Uniformer: Unifying convolution and self-attention for visual recognition. *arXiv* **2022**, arXiv:2201.09450.
25. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8759–8768.
26. Santos, T.; de Souza, L.; dos Santos, A.; Sandra, A. Embrapa Wine Grape Instance Segmentation Dataset–Embrapa WGISD. Zenodo. 2019. Available online: https://doi.org/10.5281/zenodo.3361736 (accessed on 23 June 2021).
27. Sozzi, M.; Cantalamessa, S.; Cogato, A.; Kayad, A.; Marinello, F. wGrapeUNIPD-DL: An open dataset for white grape bunch detection. *Data Brief.* **2022**, *43*, 108466. [CrossRef]
28. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
29. Li, S.; Zhang, G.; Luo, Z.; Liu, J. Dfan: Dual feature aggregation network for lightweight image super-resolution. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1–13. [CrossRef]