WEP Warwick Evans Publishing

International Journal of Computer Science and Information Technology

ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 3, Number 2, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v3n2.30
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Tomato Leaf Disease Identification Based on Yolov8

Yanli Zhong *

School of Computer Science, Yangtze University, Jingzhou, China

ABSTRACT

To quickly identify tomato leaf diseases in the planting environment, based on the YOLOv8 model, this study explores the improvement of four types of loss functions compared to the original YOLOv8 model, and replaces the backbone network of YOLOv8 to further improve the accuracy of model recognition. The improved PIoU v2 loss function has increased the Precision value by 1.1 percentage points, the Recall value by 2.8 percentage points, the mAP value by 1.3 percentage points, and the FasterNet backbone has improved the detection performance of the model, with a 0.4 percentage point increase in Precision value and a 0.3 percentage point increase in Recall value compared to the CIOU loss function used in the original model. The mAP value increased by 0.2 percentage points. Compared to the original model, the improvement effect has been improved.

KEYWORDS

Yolov8; Tomato diseases; FasterNet; PIoU v2

1. INTRODUCTION

1.1. Research Background

Tomatoes are native to South America and contain abundant carotenoids, vitamin C, and B vitamins. They have high nutritional value and can be used as both vegetables and fruits. They can also be processed to make tomato sauce and juice. They are rich in lycopene and have strong antioxidant properties, which can reduce blood pressure, clear heat, and detoxify. The United States, Russia, Italy, and China are the main producers of tomatoes.

However, tomatoes are susceptible to diseases such as tomato early blight, late blight, leaf miner, leaf mold, mosaic virus, septoria, spider mites, and yellow leaf curl virus disease. These diseases cause varying degrees of damage to the leaves, fruits, flowers, and other parts of tomatoes, affecting their yield and quality, resulting in a yield reduction rate of 20% to 30%, with some reaching over 65%, causing huge economic losses to tomato growers. Therefore, identifying and preventing tomato diseases during the growth process has great economic benefits. Traditional pest and disease monitoring and management methods often rely on manual visual inspection and processing, which is inefficient, time-consuming, and prone to missed and false detections. Therefore, utilizing computer vision and deep learning technologies to achieve automated identification and monitoring of tomato diseases and pests is of great significance.

With the continuous development and optimization of related algorithms such as vector machines, neural networks, and deep learning, the accuracy of disease and pest classification has been greatly improved. At present, machine vision technology is relatively mature and stable in the field of disease and pest detection, which can to some extent replace traditional visual recognition, reduce labor intensity, and improve detection efficiency further promoting the development of smart agriculture. The SSD [1] algorithm draws on the anchor box and regression ideas of FasterR-CNN and improves

and simplifies it, while maintaining fast speed and improving detection accuracy. Tian et al. [2] proposed an improved YOLOv3 model for detecting apples at different growth stages in orchard environments with complex backgrounds, with an F1 value of 81.7%.Mohammed Sammany et al. [3] (2007) optimized neural networks using genetic algorithms and used support vector machines to detect crop diseases. Researchers use genetic algorithms to find the optimal neural network structure and parameters to improve the performance of neural networks. In the end, unnecessary functions were successfully removed, greatly improving the detection rate and effectiveness of various diseases. Kawasak et al. [4] (2015) classified and compared normal cucumber leaves with cucumber leaves containing two common diseases based on convolutional neural networks. The model was used to identify diseases, with an average recognition accuracy of 94.9%.Habib [6] et al. proposed a papaya disease recognition system based on machine vision, using K-means clustering algorithm and SVM (Support Vector Machine) for modeling, with an accuracy of 95.2%, higher than decision trees and NB (Naive Bayes), verifying the feasibility and effectiveness of this method. El Faki et al [6]. studied a weed recognition method based on color features, which has the advantage of being less affected by factors such as shooting distance and occlusion. The experimental results showed that the method achieved recognition rates of 54.9% and 62.2% for wheat and weeds, respectively.Venugoban et al [7]. combined directional gradient histograms with Speeded up robust features to recognize and detect images of rice field diseases and pests. Shen et al. [8] applied deep neural network technology to establish a detection and recognition method for stored grain diseases and pests. Faster R-CNN was used to extract areas in the image that may contain insects, and the insects in these areas were classified with an average accuracy of 88%. Mique et al. used a CNN based model to retrieve and compare the collected images with existing rice pest and disease images, and the model achieved a final training accuracy of 90.9%. Huang et al. [9] proposed a plant disease detection method based on knowledge distillation to improve model detection performance and achieve efficient diagnosis of various crop diseases. They achieved 60.4% accuracy in the Plant Doc dataset with small model parameters mAP@.5, better than existing methods. Overall, multi-level knowledge distillation technology can make the model lighter while maintaining high accuracy.

1.2. Research Contents

In the field of tomato leaf disease identification, traditional network structures such as Google Net, ResNet, and RCNN have some drawbacks. Firstly, although Google Net and ResNet perform well on large-scale datasets such as ImageNet, they use a large number of convolutional layers and modules, and have a large number of parameters, which can easily lead to overfitting. ResNet is easily limited by network depth and has poor detection performance for small target objects. Its ability to extract detailed features of leaf diseases is limited. Their complexity and computational complexity are high, making them unsuitable for handling object detection tasks with a large number of bounding boxes. The RCNN series algorithms usually require classification and regression of a large number of candidate regions in the image, as well as separate forward inference for each candidate region, which consumes a lot of computing resources and is not suitable for real-time scenes.

Therefore, in order to address the shortcomings of traditional network structures in tomato leaf disease recognition and improve the efficiency and performance of object detection, YOLOv8 was chosen for this study. YOLOv8 is an efficient detection algorithm in the field of object detection, which adopts a dense prediction strategy to transform the object detection problem into a regression problem. The entire object detection process can be completed with only one forward propagation, and it has the advantages of fast detection speed, high accuracy, and easy deployment. By training the YOLOv8 model, eight different tomato diseases were detected and classified, and compared with traditional network structures to verify its advantages and performance in disease detection tasks. I hope that this study can provide an effective and rapid solution for the automatic identification and monitoring of tomato leaf diseases.

2. EXPERIMENTAL MATERIALS

2.1. Dataset Introduction

This dataset consists of nine categories, including eight categories of pests and diseases, and one category of healthy leaves, namely early blight, healthy, late blight, leaf miner, leaf mold, mosaic virus, septoria, spider mites, yellow leaf, curl virus, A total of 3362 images. Table 1 shows the main symptoms of tomato leaf diseases. Some tomato diseases are shown in Figure 1, and this dataset was collected from tomato leaves in a real environment, making the trained model more practical.

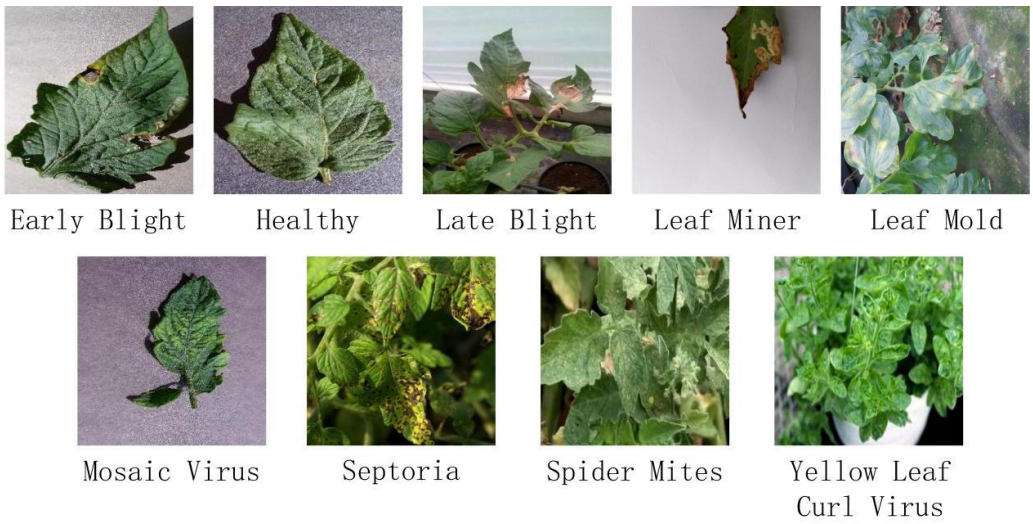

Figure 1. Illustrations of partial tomato leaf diseases

Use the LabelImg tool to label the leaves with lesions on the image, and save the annotation results in the form of category and pixel coordinates to a txt file. Divide the annotated images into training, validation, and testing sets in a 7:1:2 ratio.

Table 1. Main symptoms of tomato leaf diseases and pests

Category	Main symptoms	label
Early Blight	The leaves are damaged, initially appearing as dark brown or black circular or elliptical small spots, gradually expanding to 1-3 centimeters, with dark brown edges and gray brown centers, with concentric wheel patterns. When the weather is humid, black mold grows on the lesion, namely conidia and conidia.	Early Blight
Healthy	No lesions, appearing green or dark green.	Healthy
Late Blight	Leaf disease initially results in irregularly shaped yellow brown spots without neat boundaries. When the climate is humid, the disease spots rapidly expand, with water stains on the edges and a circle of white mold like material. On the back of the leaves, there is dense white mold growing, forming mold rings.	Late Blight
Leaf Miner	The damaged tissue of the leaves cannot grow normally, while the tissue of the other side of the leaves grows normally, causing the leaves to curl and harden.	Leaf Miner
Leaf Mold	Tomato leaf mold is an important disease of protected tomatoes, mainly affecting the leaves, stems, flowers, and fruits of tomatoes. In the early stage, some green spots appear on the back of the leaves, and in the later stage, they turn into irregular gray or black purple mold layers. The front of the leaves turns green and yellow in corresponding areas. In severe cases, the leaves often dry and curl up.	Leaf Mold
Mosaic Virus	Mosaic disease is a disease that manifests throughout the entire plant. Infected plants exhibit symptoms from the upper leaves, with alternating shades of green or mottled flowers and leaves, and severe leaf wrinkling and deformities.	Mosaic Virus
Septoria	The pathogen mainly infects the leaves. The characteristics of its lesions vary depending on the host and the type of pathogen. Generally, small brown circular spots appear on the front of the leaves initially, and gradually expand into polygonal large spots, which are gray white or light brown in color, with dark brown edges. Many small black spots are scattered or whorled within the spots.	Lead Curl
Spider Mites	Plants exhibit phenomena such as yellow leaves, scorched leaves, rolled leaves, and fallen leaves.	Spider Mites
Yellow Leaf Curl Virus	In the early stage, the main manifestations are slow or stagnant plant growth, shortened internodes, obvious dwarfing, smaller and thicker leaves, brittle and hard leaves with wrinkles, upward curling, deformation, and yellowing from the edge to the vein area of the leaves	Yellow Leaf Curl Virus

3. MATERIALS AND RESEARCH METHODS

3.1. YOLOv8 Model Structure

The YOLOv8 model supports image classification, object detection, and instance segmentation tasks. Compared to previous models (YOLOv5 and YOLOv7), it introduces a new backbone network, Darknet-53, and continues the CSP, SPPF, and PAN ideas of YOLOv5 and other architectures. It removes the convolutional structure in the PAN-FPN upsampling stage of YOLOv5, replaces the C3 module with the C2f module, and adjusts the number of channels for different scale models. The current mainstream decoupling head structure (Decoupled Head) and anchor free detection head are used to separate classification and detection heads, abandoning the previous IOU matching or

unilateral proportional allocation methods. Task Aligned Assign matching is introduced, and DFL Loss+CIOU Loss is used as the classification loss. In addition, yolov8 also uses a feature pyramid network to recognize objects of different sizes.

As shown in Figure 2, the network structure of YOLOv8 mainly consists of three major parts: Backbone, Neck, and Head. Backbone network uses a series of convolutional and deconvolution layers to extract features, as well as residual connections and bottleneck structures to reduce network size and improve performance. Neck network, including an SPPF module, a PAA module, and two PAN modules, adopts multi-scale feature fusion technology to fuse feature maps from different stages of Backbone to enhance feature representation ability. The head network head includes a detection head and a classification head. The detection head consists of a series of convolutional and deconvolution layers to generate detection results, while the classification head uses global average pooling to classify each feature map.

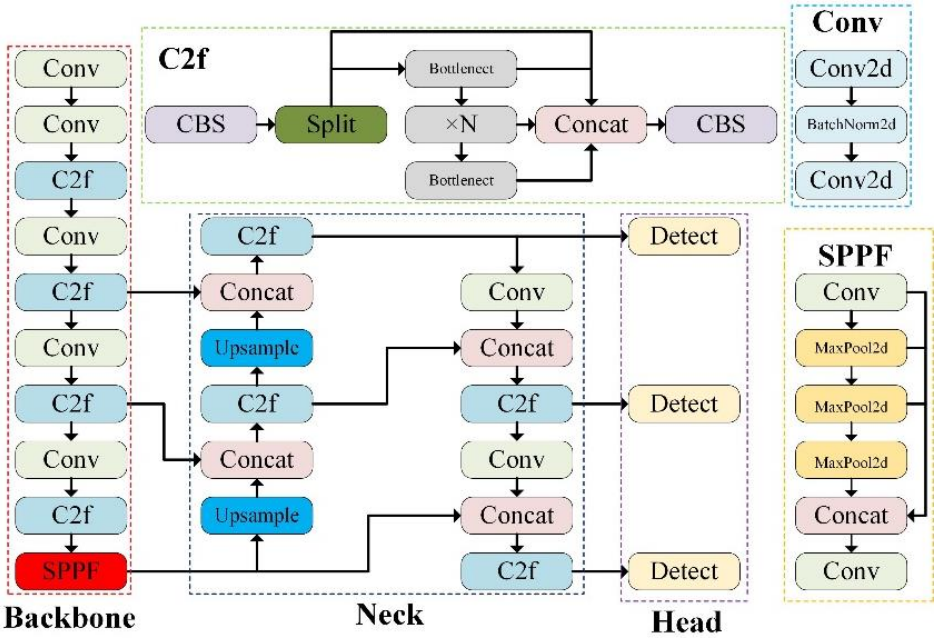

Figure 2. YOLOv8 Model Structure

3.1.1. Convolutional module conv

The convolution module conv mainly consists of a 2D convolution, a BatchNorm2d, and a SiLU activation function. Each convolution layer in the convolution module uses a stride of 2 convolution kernels for downsampling operations to reduce the size of the feature map and increase the number of channels. After the convolution layer, a Batch Normalization layer and ReLU activation function are added to enhance the nonlinear representation ability of the model.

3.1.2. Cf2 module

In the cf2 module, use a 1x1 convolutional kernel to reduce the number of input channels to half of the original, in order to reduce computational complexity and memory consumption. Use multiple 3x3 convolution kernels for convolution operations to extract feature information. Then, residual linking is used to directly add the input to the output, forming a cross layer connection. Finally, use the 1x1 convolution kernel again to recover the number of channels in the feature map.

3.1.3. SPPF

The SPPF module adopts a spatial pyramid pooling structure, and the input feature map is divided into different sub regions. The size and number of each sub region are determined by the level of pyramid pooling. Then, perform pooling operations on each subregion to average or maximize the feature maps within the subregion, and obtain the pooled features for each subregion. The pooled

features are fused together, usually through stacking or concatenation operations, to form a comprehensive feature representation. Finally, feature enhancement is carried out to further enhance the expression ability of features,

3.1.4. Concat

The Concat module mainly concatenates and fuses feature maps of different scales, transmits and interacts information of different scales to subsequent networks, introduces channel attention mechanism, and can weight feature maps based on weights

3.1.5. Upsample

The Upsample module uses bilinear interpolation to perform upsampling operations, generating more pixels and increasing the spatial size of the input feature map.

3.2. Bounding Box Loss Function

The loss function is used to evaluate the degree to which the predicted values of a model are consistent with the true values. In this study, we investigated the differences in detection efficiency between four types of bounding box loss functions, namely CIOU WIOU, EIOU, PIOUv2.

3.2.1. CIoU

Due to the fact that IoU did not consider the positional relationship and shape differences between bounding boxes, it may fail to handle situations with large size differences or angle deviations. Therefore, CIoU introduced a shape normalization factor, taking into account the aspect ratio of the bounding box in the loss function, further improving regression accuracy and being more sensitive to the position of the target box, making the loss more robust to target boxes of different shapes. The formula is as follows, where LIoU represents the intersection to union ratio between the real box and the predicted box, d represents the Euclidean distance between the center point of the real box and the center point of the predicted box, c represents the diagonal length of the minimum closure area between the real box and the predicted box, v is the parameter that measures the consistency of the aspect ratio between the real box and the predicted box, w_{gt},h_{gt},w, and h represent the width of the real box, the height of the real box, the width of the predicted box, and the height of the predicted box, respectively. α is the trade-off function for consistent aspect ratio.

$$L_{CIoU} = 1 - L_{IoU} + \frac{d^2}{C^2} + \alpha v \tag{1}$$

$$v = \frac{4}{\pi^2}(\arctan\frac{w_{gt}}{h_{gt}} - \arctan\frac{w}{h}) \tag{2}$$

$$\alpha = \frac{\mathrm{v}}{(1-L_{IoU})+v} \tag{3}$$

3.2.2. WIoU

For the detection of small and large targets, IoU may have different weights. Based on the principle of IoU, the WIoU loss function first calculates the intersection area and union area between the predicted box and the true box, introduces weight factors, and applies different degrees of weighting to different categories of targets. The weighted intersection union ratio is calculated, and its complement is taken as a negative value as the loss function, making the similarity measurement between small and large targets more balanced. The formula is as follows, where n represents the number of object boxes, b_i represents the coordinates of the i-th object box, g_i represents the coordinates of the true annotation box of the i-th object, and W_i represents the weight value

3.2.3. EIoU

IoU only considers the degree of pixel overlap between two bounding boxes, without considering the semantic information between them. EIoU represents bounding boxes by introducing embedding vectors, and uses the distance between embedding vectors to measure the similarity between bounding boxes. During the training process, the embedding vector is learned by optimizing the loss function, so that similar bounding boxes are closer in the embedding space. The formula is as follows, where b, b^{gt} are the center points of the predicted box and the true box, σ is the Euclidean distance between the two center points, c is the diagonal distance of the minimum bounding box containing A and B, h and w are the height and width of the predicted box, h^{gt} and w^{gt} are the height and width of the true box, and C_h and C_w are the height and width of the minimum bounding box.

$$L_{EIoU} = 1 - IoU + \frac{\rho^2(b,b^{gt})}{C^2} + \frac{\rho^2(w,w^{gt})}{C_w^2} + \frac{\rho^2(h,h^{gt})}{C_h^2} \tag{4}$$

3.2.4. PIoUV2

The IoU based bounding box regression loss function is easily affected by unreasonable penalty factors, leading to anchor box inflation during the regression process and slowing down convergence speed. Therefore, the PIoU loss function introduces a strong IoU loss function consisting of an adaptive penalty factor for target size and a gradient adjustment function based on anchor box quality. The loss guides anchor boxes to regress along an effective path, allowing IoU based losses to converge faster. On the basis of PIoU, PIoUv2 adds a focusing mechanism and introduces a non monotonic attention layer, which enhances the focusing ability for medium quality anchor boxes through loss enhancement. The formula is as follows, where b represents the prediction box, b 'represents the ground truth, and M represents the matching set between the prediction box and the ground truth.

$$L_{PIoU} = \frac{-\sum_{(b,b')\epsilon M} lnPIoU(b,b')}{|M|} \tag{5}$$

$$L_{PIoU\_v2} = 3(\lambda q)e^{-(\lambda q)^2} L_{PIoU} \tag{6}$$

3.3. Backbone Network

3.3.1. Fasternet

To achieve faster network speed and solve the problem of low FLOPS caused by frequent memory access of operators, partial convolutional PConV allows convolution operations to be performed on partially filled areas, more effectively extracting spatial features, reducing redundant calculations and memory access. On the basis of PConV, a new Fasttnet neural network family is proposed.

As shown in the figure3, Fasternet[10] has four stages. Stage 1 is processed using an Embedding layer (conventional Conv 4×4 with a step size of 4), while other stages are processed using a Merging layer (conventional Conv 2×2 with a step size of 2) for spatial downsampling and channel expansion. Each stage is composed of stacked FasterNet blocks, and finally, the global average pooling layer, 1×1 convolutional layer, and fully connected layer are used for feature transformation and classification.

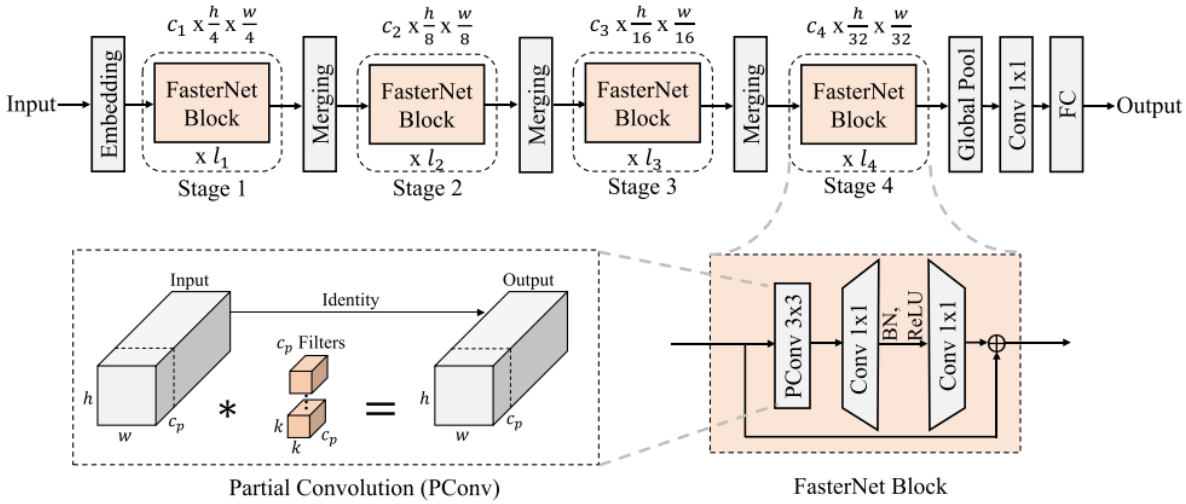

Figure 3. Working principle of PConv and FasterNet architecture

3.3.2. HGNetV2

HGNetV2 [11] is an improved and optimized version of the original HGNet model, which adopts a series of advanced neural network technologies, including Convolutional Neural Network (CNN), Residual Connection (ResNet), and multi-scale feature fusion. Through effective extraction of multi-scale features, recognition accuracy is improved. As shown in the figure 4, HGNetv2 is mainly composed of HGStem, HGBlock, and DWConv19 in the network. Data processing starts from the initial preprocessing layer of the network, HGStem, and goes through multiple HG blocks and downsampling LDS layers to reduce the dimensionality of the feature map and ultimately complete the classification task.

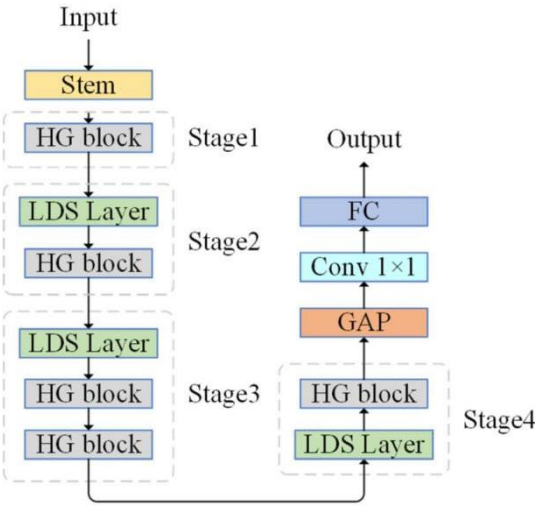

Figure 4. Structure diagram of HGNetV2

3.3.3. GhostNet

The GhostNet network achieves lightweight by reducing feature map redundancy and is composed of stacked Ghostconv modules. During the convolution process, the GhostNet module only performs a portion of the convolution operation, while the remaining portion is replaced by linear operation. The results of the two operations are concatenated to obtain a complete convolutional output. The basic operating principle of the GhostConv module is shown in the figure 5.

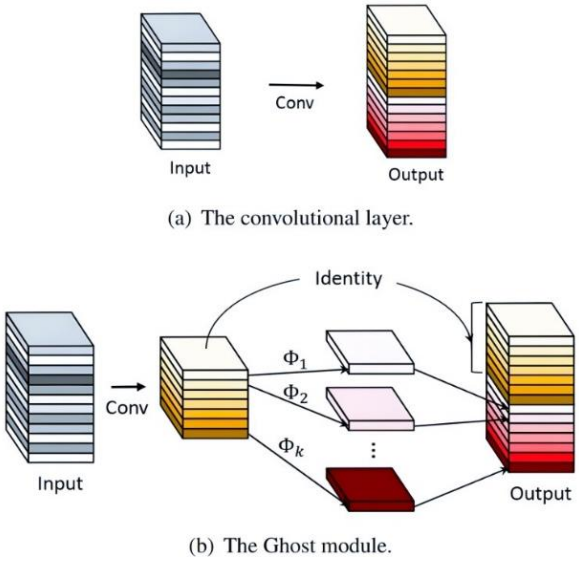

Figure 5. GhostNet structure diagram

4. EXPERIMENTAL CONTENT

4.1. Test Environment Configuration

The experiment uses Precision (P), Recall (R), Average Precision (AP), and Mean Mean Precision (mAP) to evaluate the accuracy of the detection model. The calculation formula for evaluation indicators is as follows:

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

$$Recall = \frac{TP}{TP+FN} \tag{8}$$

$$AP = \int_0^1 p(r)dr \tag{9}$$

$$mAP = \frac{1}{m}\sum_{i=1}^{m} AP_i \tag{10}$$

In the formula: *TP* represents the number of correctly recognized positive samples; *FP* represents the number of negative samples incorrectly identified as positive samples; *FN* represents the number of positive samples incorrectly identified as negative samples; *TN* represents the number of correctly recognized negative samples. *P (r)* represents a function where Precision takes Recall as a parameter; *m* represents the number of sample categories.

4.2. Evaluation Indicators

This article trains and tests custom datasets by building a PyTorch deep learning framework. The operating system is Windows 10, with a 13th Gen Intel (R) Core (TM) i5-13490F CPU, GeForce RTX 3060Ti GPU, 8GB graphics memory, 32GB memory, and training environment configured with Python 3.8, Python 1.13.1, Cuda 11.7, and YOLOv8 version 8.1.9.

4.3. Comparison Experiment of Loss Function

In order to verify the impact of various loss functions in the research method on the detection performance of the model, four bounding box loss functions, CIoU, WIoU, EIoU, and PIoUv2, were

used for comparative experiments. To ensure the reliability of the experimental results, experiments were conducted on the YOLO v8n original model. Except for the loss function, all other parameters were set the same. The results are shown in Table 2.

Table 2. The impact of different loss functions on recognition results

loss function	Precision/%	Recall/%	mAP@0.5/%	mAP50-95/%
CIoU	81.2	76.1	84.3	68.0
WIoU	83.0	76.6	84.6	68.1
EIoU	82.9	76.4	84.8	67.9
PIoUv2	82.3	78.9	85.6	68.8

From Table 2, it can be seen that the PIoU v2 loss function achieved the best performance on the tomato leaf disease detection dataset. Compared with the CIoU loss function used in the original model, the Precision value increased by 1.1 percentage points, the Recall value increased by 2.8 percentage points, and the mAP value increased by 1.3 percentage points. Although the WIoU loss function and EIoU function have slightly higher Precision values than PIoUv2, the Recall value of PIoUv2 exceeds them by two percentage points. Overall, WIoU, EIoU, and PIoUv2 can all improve the detection performance of the model compared to the CIoU bounding box loss function used in the original model, while PIoUv2 has a more significant improvement effect on this dataset.

4.4. Comparative Experiment on Backbone Network Improvement

In order to verify the impact of replacing the YOLOv8 backbone network with various network structures in the above research methods on model detection performance, all improved models were trained and tested separately. All comparative experiments were conducted using the YOLOv8n original model as the benchmark model, ensuring consistency between the dataset and other training parameters. The comparison results are shown in Table 3.

Table 3. Lightweight comparison test

Backbone network model	Precision/%	Recall/%	mAP@0.5/%	mAP50-95/%
YOLOv8n	81.2	76.1	84.3	68.0
FasterNet	81.6	76.4	84.5	68.5
HGNetV2	78.0	72.4	80.4	63.4
ghostNet	77.0	74.1	81.0	66.0

The experimental results show that after replacing the YOLOv8 backbone with the three network structures mentioned above, only the replaced FasterNet backbone improved the detection performance of the model, with a 0.4 percentage point increase in Precision value, a 0.3 percentage point increase in Recall value, and a 0.2 percentage point increase in mAP value. Compared to the YOLOv8 original model, HGNetV2 and GhostNet have significantly reduced detection performance.

5. EXPERIMENTAL CONCLUSION

This article focuses on eight common tomato diseases, including early blight, late blight, and leaf mold, as the research object. YOLOv8 is used as the basic model, and four loss functions and three backbone networks are compared on this basis. The experiments show that compared to the original YOLOv8 model, the PIoU v2 loss function has achieved the best effect on the tomato leaf disease detection dataset. The FasterNet backbone has improved the detection performance of the model, successfully improving the accuracy of tomato leaf disease recognition. This will help to identify and prevent tomato diseases. In the future, we will further explore the model structure to improve the

accuracy of tomato disease recognition. The partial experimental results using the PIOUv2 loss function are shown in Figure 6.

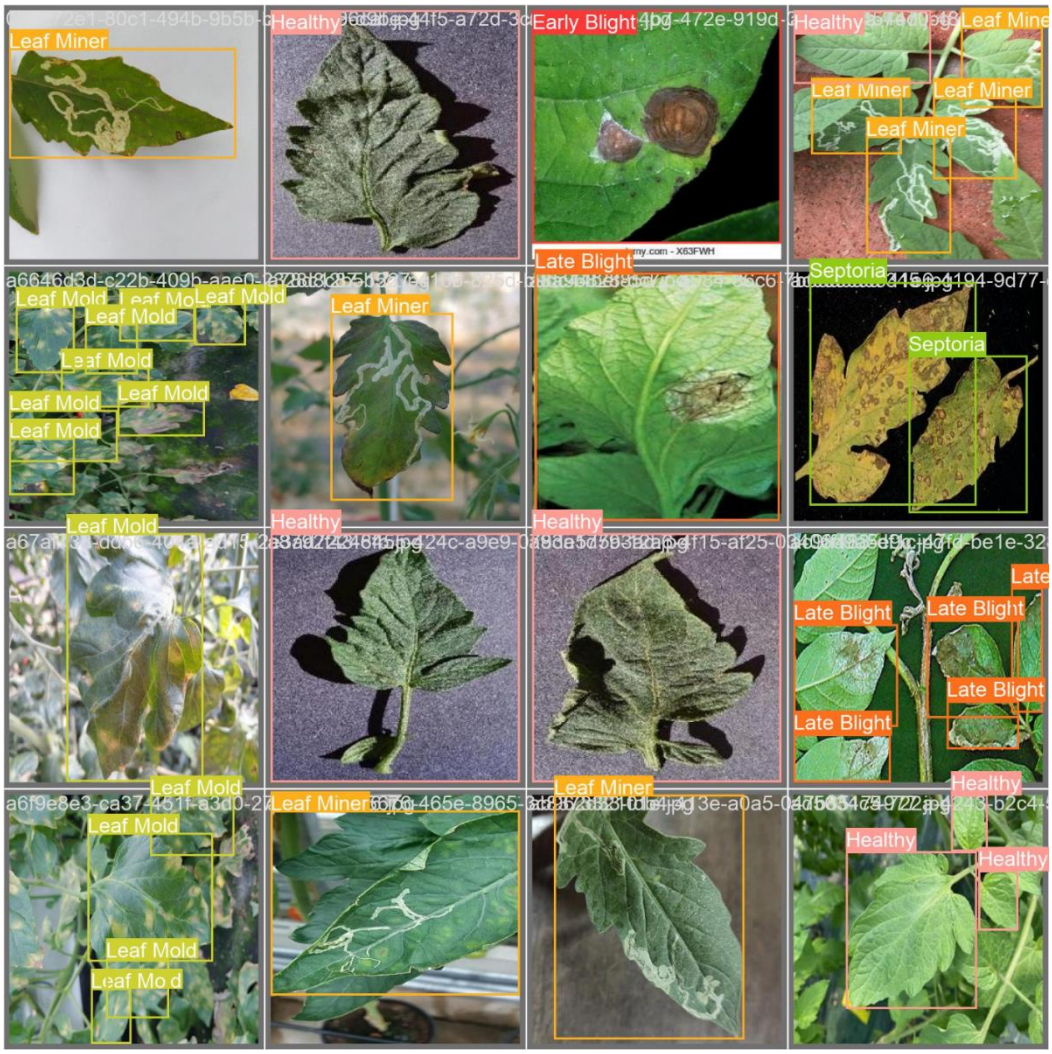

Figure 6. Partial experimental results

REFERENCES

[1] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14 (pp. 21-37). Springer International Publishing. https://doi.org/10.1007/978-3-319-46448-0_2

[2] Tian, Y., Yang, G., Wang, Z., Wang, H., Li, E., & Liang, Z. (2019). Apple detection during different growth stages in orchards using the improved YOLO-V3 model. Computers and electronics in agriculture, 157, 417-426. https://doi.org/10.1016/j.compag.2019.01.012

[3] David Story;Murat Kacira;Chieri Kubota;Ali Akoglu;Lingling An. An.Lettuce calcium deficiency detection with machine vision computed plant features in controlled environments[J].Computers and Electronics in Agriculture,2010. https://doi.org/10.1016/j.compag.2010.08.010

[4] Mahmud, M. S., Zaman, Q. U., Esau, T. J., Price, G. W., & Prithiviraj, B. (2019). Development of an artificial cloud lighting condition system using machine vision for strawberry powdery mildew disease detection. Computers and electronics in agriculture, 158, 219-225. https://doi.org/10.1016/j.compag.2019.02.007

[5] Habib, M. T., Majumder, A., Jakaria, A. Z. M., Akter, M., Uddin, M. S., & Ahmed, F. (2020). Machine vision based papaya disease recognition. Journal of King Saud University-Computer and Information Sciences, 32(3), 300-309. https://doi.org/10.1016/j.jksuci.2018.06.006

[6] El-Faki, M. S., Zhang, N., & Peterson, D. E. (2000). Weed detection using color machine vision. Transactions of the ASAE, 43(6), 1969-1978. https://doi.org/10.13031/2013.3103

[7] Vorugunti, C. S., Pulabaigari, V., Gorthi, R. K. S. S., & Mukherjee, P. (2020). Osvfusenet: online signature verification by feature fusion and depth-wise separable convolution based deep learning. Neurocomputing, 409, 157-172. https://doi.org/10.1016/j.neucom.2020.05.072

[8] SHEN Y,ZHOU H,LI J,et al.Detection of stored-grain insects using deep learning[J].Computers and Electronics in Agriculture,2018,319-325. https://doi.org/10.1016/j.compag.2017.11.039

[9] Huang, Q., Wu, X., Wang, Q., Dong, X., Qin, Y., Wu, X., ... & Hao, G. (2023). Knowledge distillation facilitates the lightweight and efficient plant diseases detection model. Plant Phenomics, 5, 0062. https://doi.org/10.34133/plantphenomics.0062

[10] Chen, J., Kao, S. H., He, H., Zhuo, W., Wen, S., Lee, C. H., & Chan, S. H. G. (2023). Run, Don't walk: Chasing higher FLOPS for faster neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 12021-12031). https://doi.org/10.1109/CVPR52729.2023.01157

[11] Vorugunti, C. S., Pulabaigari, V., Gorthi, R. K. S. S., & Mukherjee, P. (2020). Osvfusenet: online signature verification by feature fusion and depth-wise separable convolution based deep learning. Neurocomputing, 409, 157-172. https://doi.org/10.1016/j.neucom.2020.05.072