Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.060550

ARTICLE

LT-YOLO: A Lightweight Network for Detecting Tomato Leaf Diseases

Zhenyang He and Mengjun Tong*

College of Mathematics and Computer Science, Zhejiang Agricultural and Forestry University, Hangzhou, 310000, China

*Corresponding Author: Mengjun Tong. Email: 20130035@zafu.edu.cn

Received: 04 November 2024; Accepted: 12 December 2024; Published: 06 March 2025

ABSTRACT: Tomato plant diseases often first manifest on the leaves, making the detection of tomato leaf diseases particularly crucial for the tomato cultivation industry. However, conventional deep learning models face challenges such as large model sizes and slow detection speeds when deployed on resource-constrained platforms and agricultural machinery. This paper proposes a lightweight model for detecting tomato leaf diseases, named LT-YOLO, based on the YOLOv8n architecture. First, we enhance the C2f module into a RepViT Block (RVB) with decoupled token and channel mixers to reduce the cost of feature extraction. Next, we incorporate a novel Efficient Multi-Scale Attention (EMA) mechanism in the deeper layers of the backbone to improve detection of critical disease features. Additionally, we design a lightweight detection head, LT-Detect, using Partial Convolution (PConv) to significantly reduce the classification and localization costs during detection. Finally, we introduce a Receptive Field Block (RFB) in the shallow layers of the backbone to expand the model's receptive field, enabling effective detection of diseases at various scales. The improved model reduces the number of parameters by 43% and the computational load by 50%. Additionally, it achieves a mean Average Precision (mAP) of 90.9% on a publicly available dataset containing 3641 images of tomato leaf diseases, with only a 0.7% decrease compared to the baseline model. This demonstrates that the model maintains excellent accuracy while being lightweight, making it suitable for rapid detection of tomato leaf diseases.

KEYWORDS: YOLOv8n; target detection; lightweight; tomato; attention mechanism

1 Introduction

Tomatoes, as a widely cultivated fruit and vegetable, have an annual global production exceeding 170 million tons, often ranking first among vegetable crops [1]. Tomato cultivation in China has a history of nearly 100 years [2]. China ranks first in global fresh tomato production and typically ranks second or third in processed tomato production. Despite challenges such as an aging population and labor shortages, tomato farming in China still relies heavily on manual labor. This increases the intensity of labor, worsens working conditions, and raises both time and labor costs. Additionally, tomatoes are vulnerable to various diseases during growth, which not only hampers their development but also causes significant agricultural economic losses [3]. Among these diseases, the tobacco mosaic virus (TMV) is a major pathogen affecting tomato leaves and accounts for a large proportion of disease burden [4]. Since most tomato diseases first appear on the leaves and then spread to the entire plant, they severely impede the normal growth of tomato plants, sometimes even halting growth entirely. Therefore, timely detection and identification of tomato leaf diseases are crucial for the precise application of pesticides and controlling disease spread.

Object detection is fundamental to various computer vision tasks such as instance segmentation, keypoint detection, and object tracking. In recent years, the rapid advancement of deep learning technology [5] has significantly driven progress in object detection, leading to remarkable breakthroughs and making it a prominent research focus. Object detection is now widely applied in agriculture, including tasks like automated harvesting, pesticide spraying, and plant disease detection. Among the object detection algorithms, YOLO (You Only Look Once) stands out for its high speed and accuracy. This paper selects the YOLOv8 series, a novel and stable version of the YOLO family, for tomato leaf disease detection. However, due to the specific requirements of this task, the model's deployment environment is often constrained by limited computational power, particularly for real-time detection on resource-constrained platforms. This presents a challenge for model light-weighting. Furthermore, the size disparity of different disease areas on the leaves, with some diseases being too small for accurate detection, adds to the difficulty. Variable outdoor lighting conditions during detection also pose challenges. Due to these factors, the base YOLOv8 model struggles to meet the required standards in tomato leaf disease detection, as its size and detection speed are insufficient. Therefore, further research and optimization are needed.

To address these issues, this paper proposes a lightweight object detection model, LT-YOLO (Light Tomato-YOLO), for high-performance detection of tomato leaf diseases in a lightweight model. The main contributions of this study are as follows:

1. The RVB (RepViT Block) [6] is introduced for feature extraction to achieve a lightweight design in feature extraction.

2. In the feature extraction backbone, the EMA (Efficient Multi-Scale Attention) [7] mechanism is applied to certain blocks. This allows the model to maintain the channel dimensions while learning effective channel descriptions and generating better pixel-level attention for producing high-level feature maps.

3. For the object classification head, a lightweight detection head is designed by integrating the Partial Convolution (PConv) [8]. This reduces computational complexity and operational costs significantly, albeit with a slight decrease in accuracy.

4. In the shallow layers of feature extraction, the Receptive Field Block (RFB) [9] is added to expand the receptive field and address the detection of diseases of varying sizes.

2 Related Work

Object detection techniques based on handcrafted feature extraction faced significant bottlenecks around 2010. With the maturation of deep learning technologies, object detection methods based on Convolutional Neural Networks (CNNs) were first proposed in 2014 [10,11]. Since then, CNN-based object detection techniques have advanced rapidly and are now broadly divided into two categories: multi-stage object detection and single-stage object detection. Multi-stage models follow a coarse-to-fine detection process, initially improving recall rates and then refining the localization for greater precision, typically achieving higher accuracy. In contrast, single-stage object detection models detect all objects in one inference step, offering superior speed.

2.1 Multi-Stage Object Detection

Prominent examples of multi-stage object detection include RCNN, Faster-RCNN [12], and SPPNet [13]. Significant progress has been made in optimizing multi-stage object detection models. For instance, Peng et al. [14], building on Faster-RCNN, introduced a multi-network adaptive distillation method and designed an efficient, end-to-end incremental object detection model. This approach addresses the problem of catastrophic forgetting [15] in deep learning models, enabling the model to learn new knowledge without forgetting previously acquired knowledge. Ren et al. [16] replaced the fully connected layers in Fast/Faster-RCNN's classification head with convolutional layers. Their experiments demonstrated that using a deeper

ConvNet as the feature classifier enhances detection performance. Sharma et al. [17] based on Faster-RCNN, utilized saliency detection, proposal generation, and bounding box regression to improve detection and loss functions, proposing SGFr-RCNN, which achieved higher mean accuracy on public datasets such as PASCAL VOC 2007. Wang et al. [18] replaced the Visual Geometry Group Network (VGG) backbone in the Faster R-CNN model with Res2Net101 to enhance the receptive field representation capability of each network layer. This modification improved the network's mAP@0.5 to 71.7%, representing a 3.3% increase compared to the original Faster R-CNN model.

2.2 Single-Stage Object Detection

YOLO [19], SSD (Single Shot MultiBox Detector) [20], and RetinaNet [21] are exemplary single-stage object detection models. Zhang et al. [22] proposed a lightweight underwater object detection method based on YOLOv4, which incorporates MobileNetV2 and depthwise separable convolution to reduce the number of model parameters and the overall model size. The method integrates attentional feature fusion to achieve lightweight target detection in marine environments. Cui et al. [23] replaced the backbone network of YOLOv4-Tiny with the lightweight Enhance ShuffleNet to extract pinecone features. By introducing a squeeze-and-excitation feature pyramid network for multi-scale information fusion and retaining only a 26×26 detection head for pinecone prediction, the computational cost was reduced to 17.35% of YOLOv4-Tiny. Li et al. [24] designed a mobile-based Apple-YOLO model by incorporating the double-branch Apple-CSP module and a Focus layer with depthwise separable convolution and an attention mechanism module into YOLOv5, effectively reducing the model's Floating Point Operations (FLOPs) and achieving an mAP of 96.04%. Wang et al. [25] pruned the convolution kernels of the YOLOv4-tiny model and integrated dilated convolution layers into the residual module. Additionally, they introduced the Receptive Field Block (RFB) module, which mimics the human visual receptive field to expand the model's receptive field and enhance feature extraction efficiency. By combining spatial and channel attention via the convolutional block attention module, they further improved the model's ability to capture effective features and reduced the adverse impact of noise on performance. Zhai et al. [26] addressed SSD's lack of feature complementarity between feature layers and its poor performance in small object detection by proposing an improved SSD algorithm based on DenseNet and feature fusion, increasing detection accuracy by 3.1% mAP on VOC 2007. Hu et al. [27] selected a scaling factor to lightweight the YOLOv7 model. By integrating ECA and CA attention mechanisms with ELAN-B3 and DownC modules, they developed a new lightweight model called Multimodule-YOLOv7-L, which provides valuable insights for designing intelligent robots for inter-row weed control. Ren et al. [28] designed a lightweight feature enhancement backbone network (LFEBNet) to reduce computational costs and constructed a Channel and Position Enhanced Attention (CPEA) module. By embedding the CPEA module into the backbone network, they effectively captured positional information to more accurately locate target positions. Based on YOLOv5, they proposed an efficient lightweight network called YOLO-Lite for SAR ship detection.

In this study, we selected the YOLOv8 model for the following reasons: 1. For lightweight object detection tasks, single-stage object detection models inherently outperform multi-stage models in terms of model size and detection speed. 2. The YOLO series of single-stage object detection models offers comprehensive documentation and strong community support, which greatly facilitates model usage, debugging, and deployment, thereby aiding the execution of our experiments. 3. YOLOv8 has been extensively validated through practical applications, demonstrating its reliability and stability.

3 Proposed Methods

3.1 YOLOv8n

The basic structure of YOLOv8n is illustrated in Fig. 1. Aside from the input layer, the YOLOv8n object detection model is primarily composed of three sections: the BackBone, which extracts feature information; the Neck, which facilitates feature fusion and interaction across different levels; and the Head, responsible for object localization and classification. The Conv module consists of standard convolution blocks, batch normalization layers, and the SiLU activation function. The Conv block is typically used to apply convolution to input images or feature maps, reducing resolution while increasing the number of channels. The C2f module is designed for deep feature extraction. It preserves the size of the input and output feature maps, while performing lightweight feature extraction through its internal Bottleneck residual structure. Finally, the Head layer uses three decoupled detection heads corresponding to feature maps of different scales, outputting the predicted object locations and classification information.

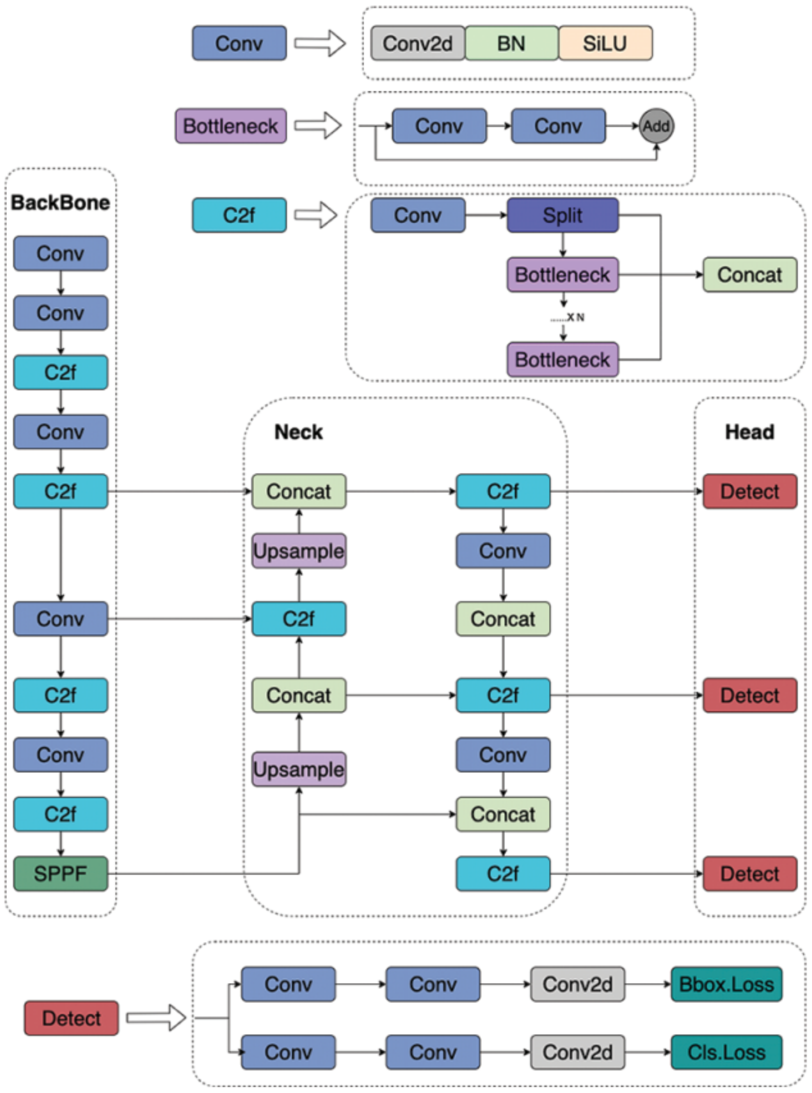

Figure 1: The YOLOv8n structural model consists of three main components: Backbone, neck, and head

3.2 Improvements to YOLOv8n

To address the challenges encountered by the basic YOLOv8n model in tomato leaf disease detection, and to meet the lightweight requirements for agricultural applications, we developed a lightweight LT-YOLO object detection network. As shown in Fig. 2, this algorithm includes four key improvements. First, we replaced the C2f feature extraction block with the C2fRVB, achieving better feature extraction performance with the same number of parameters and computational complexity. Second, we applied the EMA attention mechanism to the deeper layers of the feature extraction network to improve the capture of key feature maps with minimal computational cost, we refer to it as C2fRVBE.Additionally, we significantly reduced the size of the detection heads using PConv (partial convolution), resulting in only a slight loss in accuracy. Finally, we incorporated the RFB module into the shallow layers of the feature extraction network to expand the model's receptive field, enabling better detection of diseases across various sizes.

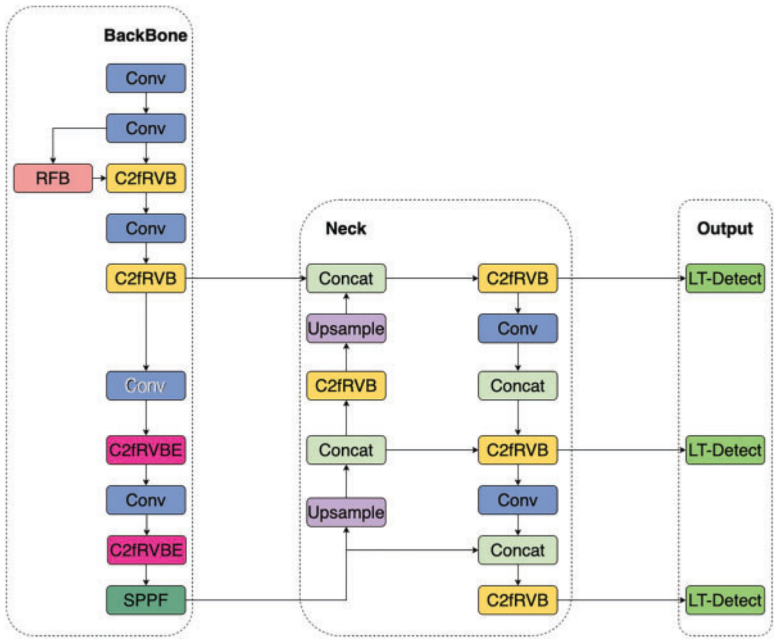

Figure 2: Improved YOLOv8n model

3.3 RepViT Block

Recently, lightweight vision transformers (ViTs) have demonstrated superior performance and lower latency compared to lightweight convolutional neural networks (CNNs) on resource-constrained mobile devices [29]. The block structures of lightweight ViTs typically include separate token mixers and channel mixers [30]. Building on this insight, the RVB (RepViT Block) splits the channel mixer and token mixer found in the original MobileNetV3 block to emulate the behavior of existing lightweight ViTs.

As depicted in Fig. 3a, the original MobileNetV3 block employs 1 × 1 expansion convolutions and 1 × 1 projection convolutions to facilitate channel interaction, effectively serving as a channel mixer. Subsequently, it utilizes depthwise (DW) convolutions following the 1 × 1 expansion convolution to amalgamate spatial information, acting in the capacity of a token mixer. This configuration integrates the channel mixer with the token mixer. Within the RVB block, the depthwise convolution is repositioned upwards, and the SE (squeeze-and-excitation) block, which is predicated on spatial data, is also relocated prior to the expansion convolution. Ultimately, structural reparameterization techniques [31] are applied to the DW layer to augment the model's learning capability without incurring additional computational expenses during inference.

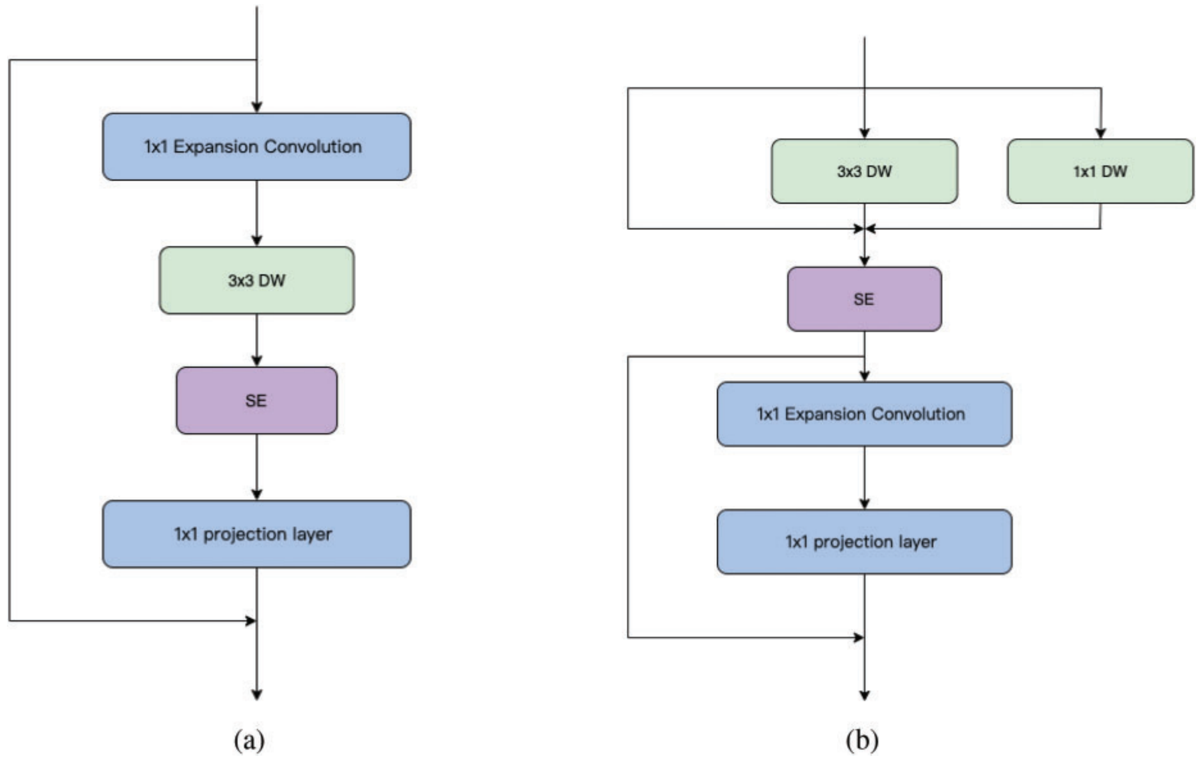

Figure 3: (a) Original MobileNetV3 block; (b) RepViT block

3.4 Efficient Multi-Scale Attention

There are three widely recognized attention mechanisms: channel attention, spatial attention, and a combination of both [7]. EMA (Efficient Multi-Scale Attention) is inspired by attention mechanisms such as SE (Squeeze-and-Excitation) [32], CBAM (Convolutional Block Attention Module) [33], and CA (Coordinate Attention) [34], observing that cross-dimensional interaction aids in predicting channel or spatial attention. Based on the fundamental structure of the CA block, the design of the EMA block is illustrated in Fig. 4.

Compared to CA, EMA introduces a unique cross-spatial learning capability, offering a method for cross-spatial information aggregation along different spatial dimensions, leading to richer feature aggregation. It not only encodes inter-channel information to adjust the importance of different channels but also retains precise spatial structure information within the channels. As shown in Fig. 5, we replaced the SE block in the original RVB with EMA attention, creating the C2fRVBE module. To reduce additional computational overhead, EMA is applied only to the deeper layers of the BackBone.

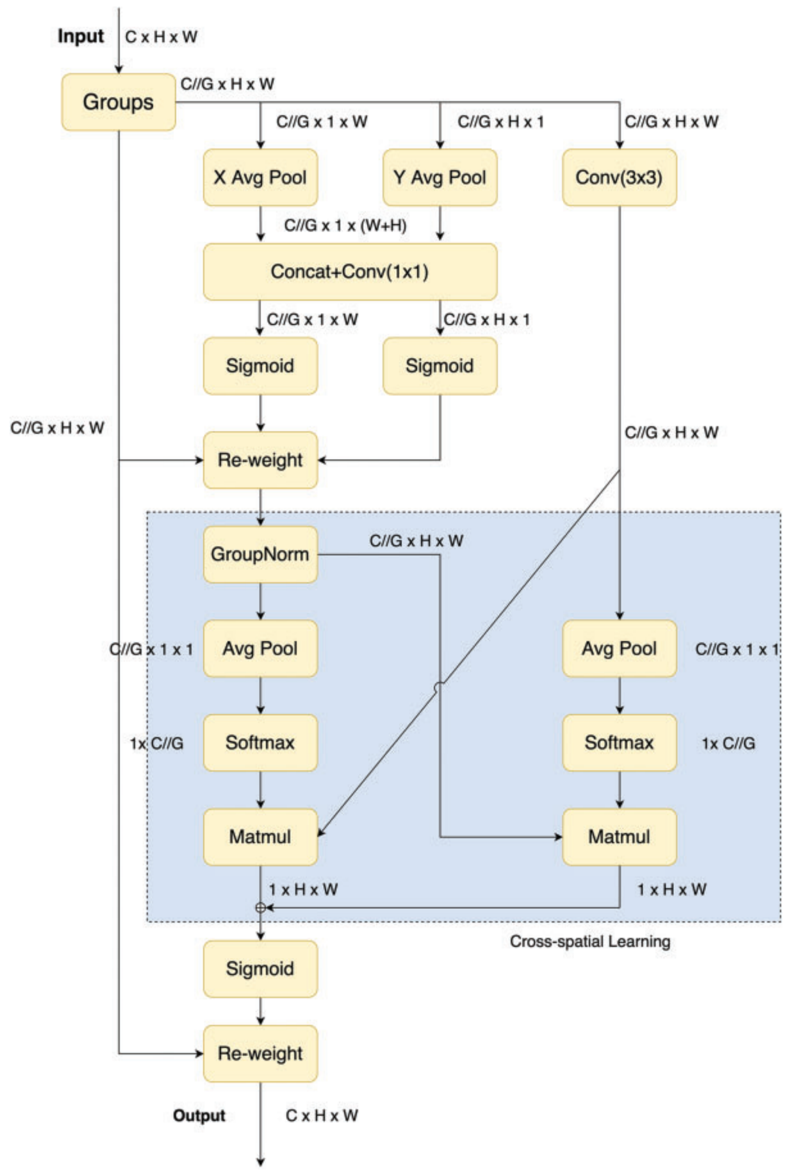

Figure 4: The specific structure of the EMA block, with the blue box indicating its unique cross-spatial learning component

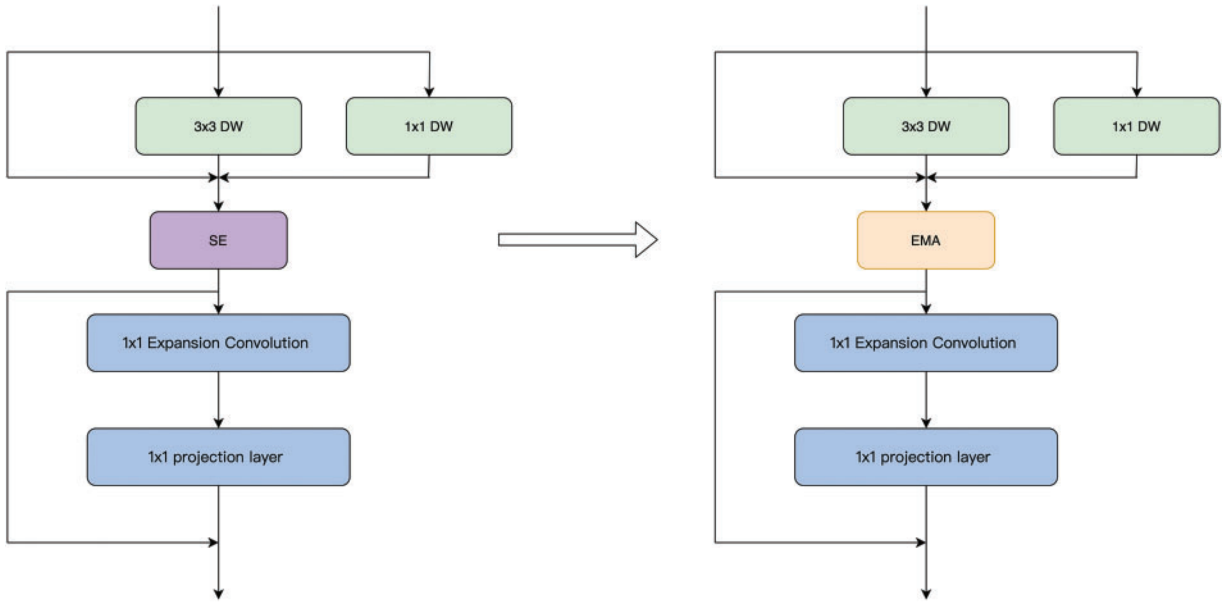

Figure 5: Incorporating EMA into RepViT Block

3.5 Lightweight Detection Head Based on PConv

In neural network training, a key factor affecting network speed, apart from FLOPs (Floating Point Operations), is FLOPS (Floating Point Operations Per Second), which measures effective computational speed. Generally, network latency is related to both FLOPs and FLOPS. A primary cause of low FLOPS is frequent memory access [8].

$$Latency = \frac{FLOPs}{FLOPS} \tag{1}$$

To address this issue, PConv (Partial Convolution) was proposed. PConv reduces both FLOPs and memory access, thereby increasing the convolutional operation's FLOPS. In convolutional neural networks, feature maps across different channels often exhibit high redundancy [8]. However, few approaches effectively exploit this redundancy in a simple and efficient manner.

As illustrated in Fig. 6, PConv applies standard convolution to only a portion of the input feature map's channels, leaving the remaining channels unchanged. Typically, PConv selects the first c_p channels to represent the entire feature map. Record the input height as h, width as w, and convolution kernel size as k, the FLOPs of a PConv operation are given by:

$$h \times w \times k^2 \times c_p^2 \tag{2}$$

Record the number of input channels as c, under a compression ratio of $r = \frac{c_p}{c} = \frac{1}{4}$, PConv's FLOPs are only $\frac{1}{16}$ of those of a standard convolution, and its memory access requirement is similarly reduced. It requires only $\frac{1}{4}$ of the memory access compared to a standard convolution [8].

As shown in Fig. 7, YOLOv8n employs a decoupled detection head, where Localization loss (Bbox.Loss) and Classification (Cls.Loss) are computed separately, and their results are summed as the final loss. Each part of the decoupled head consists of two 3×3 convolutional blocks and a final 1×1 convolution.

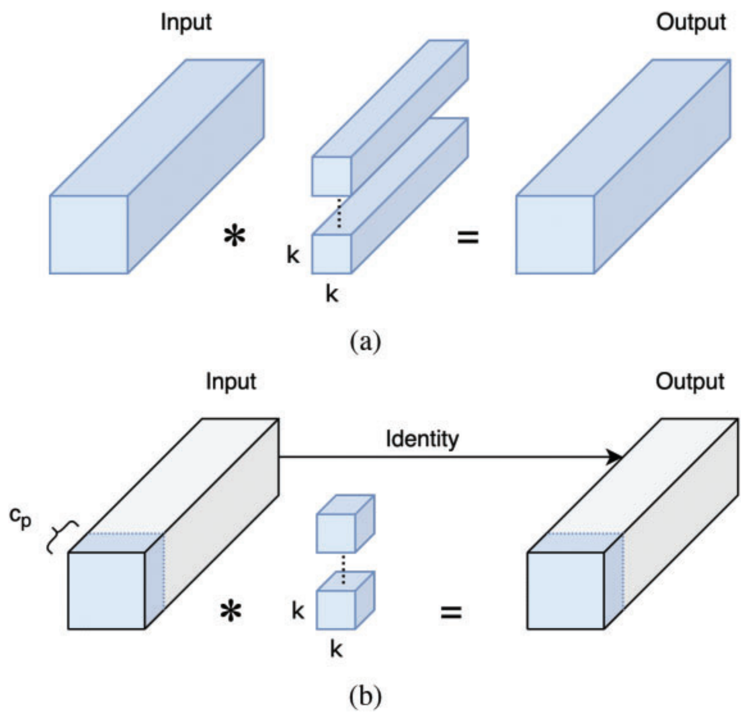

Figure 6: (a) Conventional convolution; (b) Partial convolution

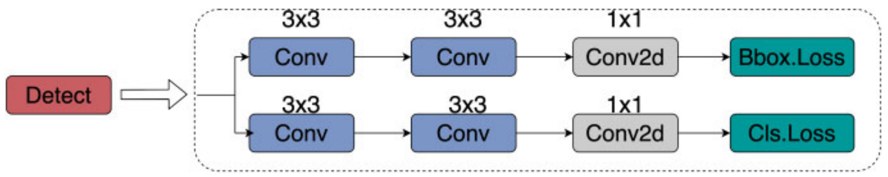

Figure 7: YOLOv8 detection head

Our improved detection head is shown in Fig. 8. In the initial stage of the decoupled head, we applied PConv with a 3 × 3 kernel and a compression ratio. This is followed by a 1 × 1 convolutional block and then a standard 1 × 1 convolution. We conducted comparison experiments to select the optimal value for the compression r, and results showed that the improved lightweight detection head significantly reduces the number of parameters and computational cost with only minimal loss in accuracy.

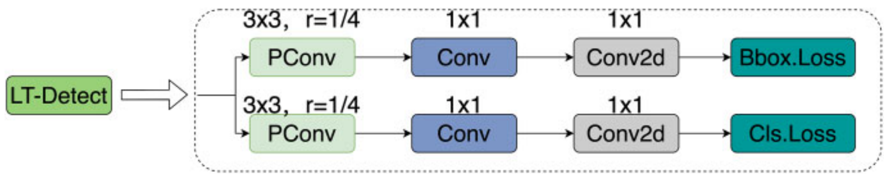

Figure 8: LT-detect detection head

3.6 Receptive Field Block

The Receptive Field Block (RFB) is a novel module inspired by the structure of receptive fields (RFs) in the human visual system. It was proposed to enhance the distinctiveness and robustness of features by considering the relationship between the size of the receptive field and its eccentricity.

As shown in Fig. 9, the RFB employs a multi-branch structure that combines different sizes of convolutional kernels and dilated convolution layers. This design simulates the human visual system's ability to focus on objects of varying sizes, thereby expanding the receptive field of the model. This not only increases the receptive field size but also strengthens the detection capability for small disease spots, effectively extracting features of different sizes.

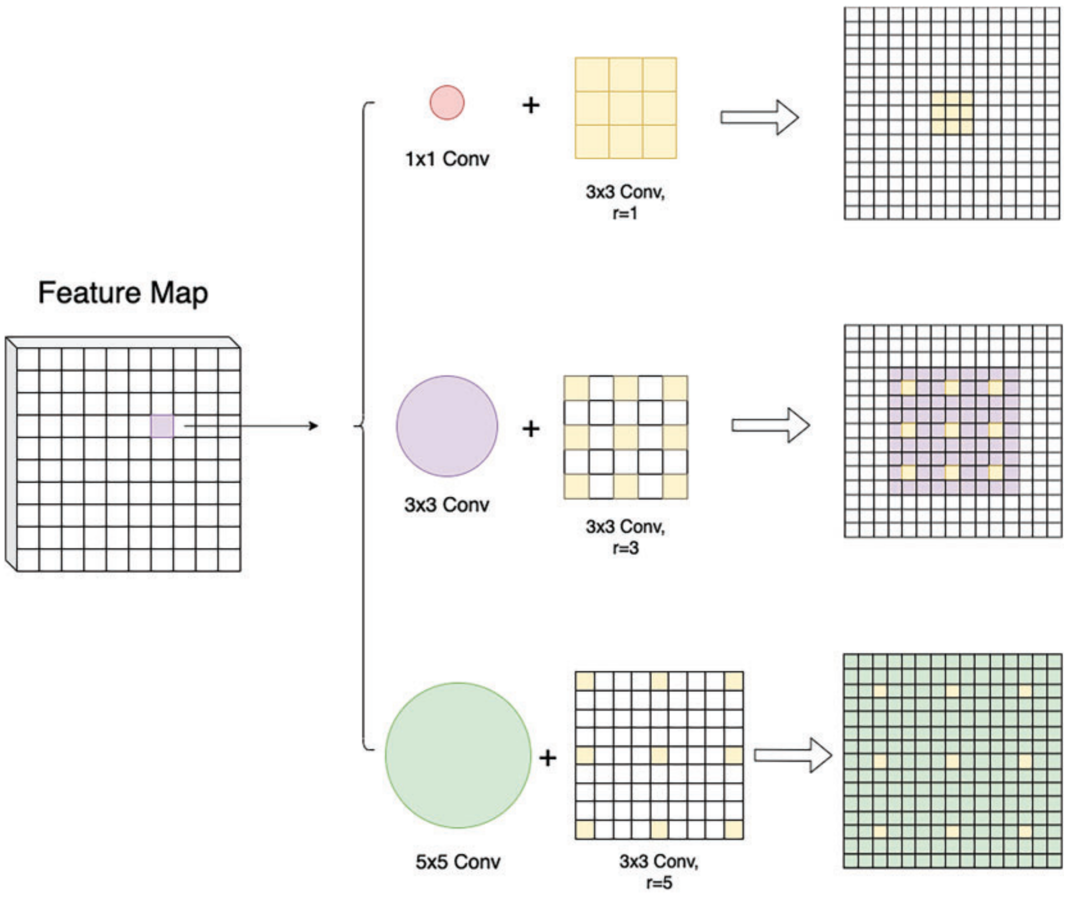

Figure 9: Multi-branch receptive field of the receptive field block

As Fig. 10, the RFB we utilized includes three branches with dilated convolutions, with dilation rates of 3, 3, and 5, respectively. In the fourth branch, a standard 3 × 3 convolution is used to refine the extraction of feature details, and its results are concatenated with the outputs of the other convolutions. Lastly, a residual connection is applied in the fifth branch.

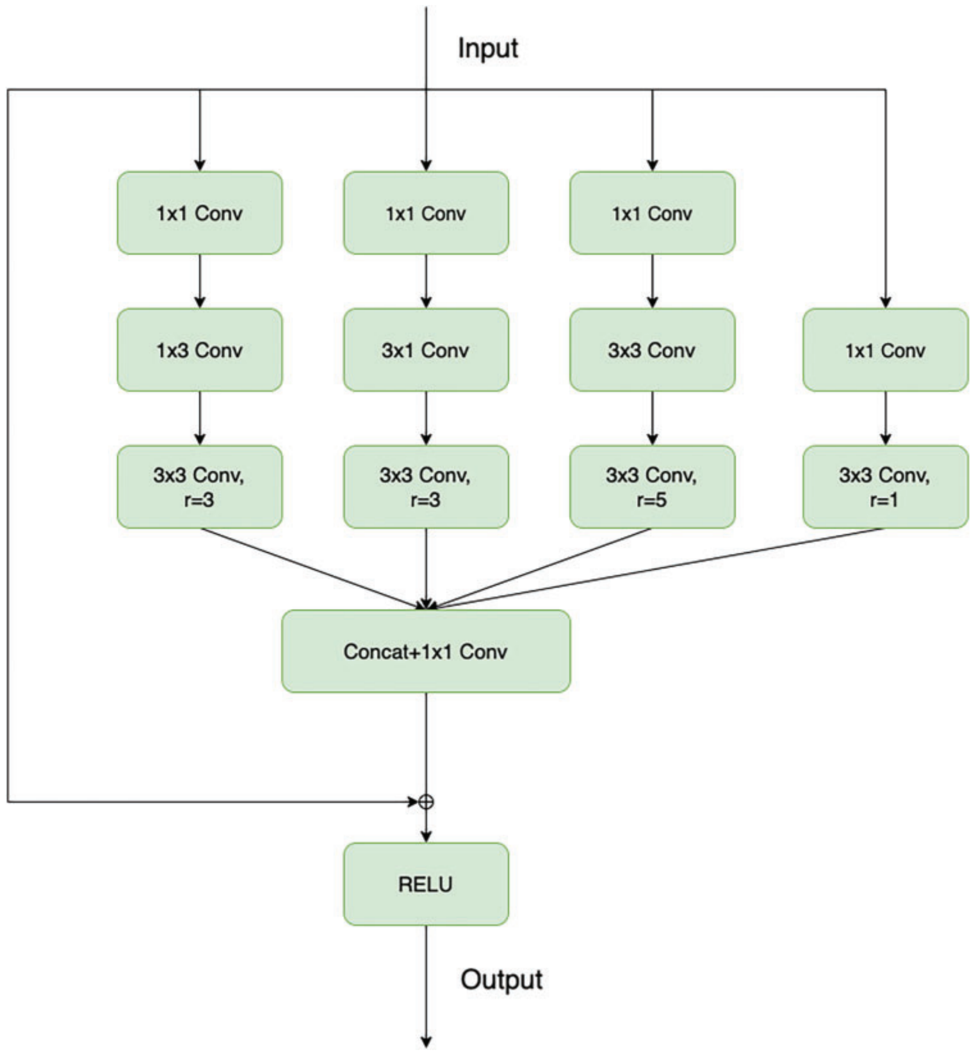

Figure 10: The specific structure of receptive field block

4 Experiment

In these experiments, we used a server equipped with an Intel Platinum 8255C CPU (2.50 GHz) and an RTX 3080 (10 GB) GPU as the experimental environment. The operating system was Linux Ubuntu 20.04. Both the training and inference phases used an input resolution of 448 × 448, with a batch size set to 16. The training was conducted for a total of 500 epochs, with a patience parameter of 40, meaning the training would stop if no improvement in accuracy was observed for 40 consecutive epochs. The chosen optimization algorithm was SGD (Stochastic Gradient Descent), with an initial learning rate of 0.01.

4.1 Dataset

The dataset used in this study comes from Roboflow's open-source pest and disease dataset [35]. The original dataset contains nine categories related to tomato leaf pests and diseases, including eight disease/pest categories and one healthy category. To focus specifically on tomato leaf diseases, we filtered out the pest categories, leaving six disease categories and one healthy category. The diseases included are Early Blight, Late Blight, Yellow Leaf Curl Virus, Leaf Mold, Mosaic Virus, and Septoria. In total, 3641 images were used, including the healthy leaf category. The dataset was split into training, validation, and testing sets at a ratio of 8:1:1. The six diseases in the dataset are shown in the Fig. 11.

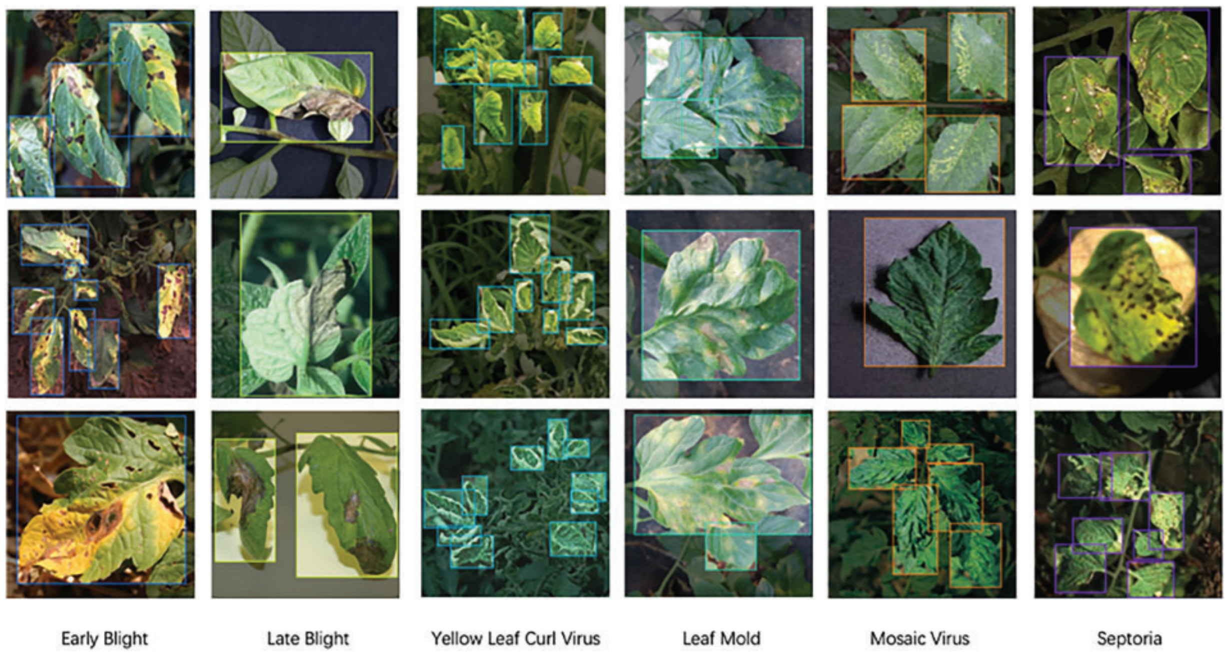

Figure 11: Six types of tomato leaf diseases

4.2 Evaluation Metrics

The primary metrics used to evaluate model size in this study are Parameters(Params) and Floating Point Operations (FLOPs). Parameters is a key indicator of spatial complexity, directly impacting model efficiency. Floating Point Operations represent the model's temporal complexity, determining how long it takes the network to complete tasks.

Additionally, precision, recall, and mAP@0.5 are selected because they are widely used metrics in the object detection domain, effectively evaluating the model's detection accuracy and recall ability. The selection of evaluation metrics should align with the study's specific goals and challenges. Our focus on lightweight and high-speed detection justifies the use of these metrics. Precision is the proportion of correctly predicted positive samples among all samples predicted as positive, reflecting the model's accuracy. Precision is calculated as:

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

where TP (True Positive) is the number of correctly predicted positive samples, and FP (False Positive) is the number of incorrect positive predictions. Recall is the proportion of correctly predicted positive samples out of all actual positive samples, reflecting the model's ability to capture positive cases. Recall is calculated as:

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

where FN (False Negative) is the number of incorrectly predicted negative samples.

$$AP@0.5 = \frac{1}{n}\sum_{i=1}^{n} P_i = \frac{1}{n}P_1 + \frac{1}{n}P_2 + \cdots + \frac{1}{n}P_n \tag{5}$$

$$mAP@0.5 = \frac{1}{N}\sum_{k=1}^{N} AP@0.5_k \tag{6}$$

AP@0.5 refers to the Average Precision for a specific category when the Intersection over Union (IoU) threshold is set to 0.5. In object detection, IoU measures the overlap between two bounding boxes, calculated as the ratio of their intersection to their union. AP@0.5 means that a prediction is considered correct if the IoU between the predicted and ground truth boxes is at least 0.5. The mAP@0.5 (mean Average Precision at IoU 0.5) is the average AP@0.5 across all categories.

4.3 Comparative Experiments

To further evaluate the performance of LT-YOLO, we compared it with seven representative models in the field of object detection: Faster R-CNN, SSD, RT-DETR, YOLOv5n, YOLOv8n, YOLOv10n, and YOLO11n. The evaluation focused on two main aspects: model size and detection capability. As shown in Table 1.

Table 1: Model performance comparison

Model	P(%)	R(%)	mAP@0.5(%)	Params(M)	FLOPs(G)
Faster-RCNN	91.5	81.6	90.1	137.1	370.2
SSD	90.2	81.7	91.4	24.5	30.7
RT-DETR(L)	91.5	78.2	86.9	26.6	51.1
YOLOv5n	88.4	82.1	89.5	2.5	3.5
YOLOv8n	90.8	81.8	91.6	3.0	4.0
YOLOv10n	88.1	80.5	90.3	2.7	4.0
YOLO11n	89.5	82.9	91.7	2.6	3.1
LT-YOLO(Ours)	88.1	83.4	90.9	1.7	2.0

The experimental results demonstrated that the LT-YOLO model has significantly fewer parameters and lower computational complexity than the other models, with counts of 1.7 M parameters and 2.0 G FLOPs, thereby meeting the lightweight requirements. Compared to the baseline model YOLOv8n, LT-YOLO reduces parameters by approximately 43.3% and FLOPs by 50%, while only sacrificing 0.7% in mAP@0.5. Furthermore, LT-YOLO shows an improvement of 1.6% in recall. When compared to the latest second-best model, YOLO11n, LT-YOLO decreases mAP by 0.8%, along with reductions of 34.6% in parameters and 35.5% in computational complexity, while still maintaining excellent performance.

4.4 Training Process Analysis

Additionally, we tracked the training processes of both the improved model and the baseline model as Fig. 12. The experiments revealed that the training of the baseline model, YOLOv8n, stopped at 314 epochs, achieving convergence at 274 epochs. In contrast, the LT-YOLO model trained for 318 epochs, with convergence occurring at 278 epochs. This indicates that the modifications made to the model did not result in a significant increase in training duration.

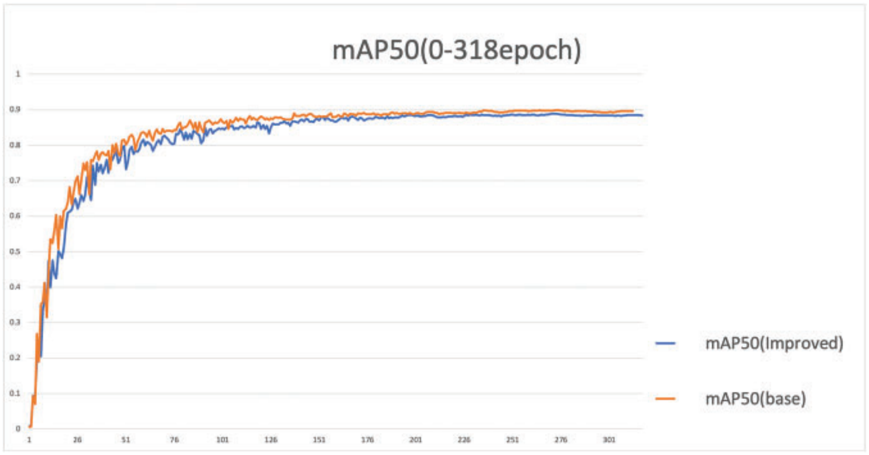

Figure 12: The mAP during training process of the base model and the improved model

4.5 *Ablation Study*

To assess the effectiveness of each improvement module, we conducted a series of experiments by systematically combining the modules. The primary metrics for the ablation study included Precision, Recall, mAP@0.5, Params, and FLOPs. As shown in Table 2.

Table 2: Ablation experiment results

RVB	EMA	LT-Detect	RFB	P(%)	R(%)	mAP@0.5	Params(M)	FLOPs(G)
				90.8	81.8	91.6	3.0	4.0
✓				89.8	81.7	90.4	2.3	3.1
✓	✓			90.1	82.8	90.9	2.3	3.2
		✓		88.5	82.6	90.1	2.4	2.7
			✓	90.6	83.5	91.8	3.1	4.1
✓	✓	✓		88.4	82.2	90.7	1.7	1.9
✓	✓	✓	✓	88.1	83.4	90.9	1.7	2.0

From the table, it is evident that each of the designed modules contributes to either an increase in mAP@0.5 or a reduction in parameters and computational load. The first row represents the performance of the baseline model, YOLOv8n, without any modifications. Replacing the C2f module with RVB resulted in an overall reduction of approximately 23.3% in parameters and 22.5% in computational load. The integration of the EMA attention mechanism within the RVB module and its application to deeper layers improved mAP@0.5 by 0.5%. Utilizing LT-Detect for the lightweight detection head led to a 20% decrease in parameters and a 32.5% decrease in computational load. Additionally, the incorporation of RFB in the shallow feature extraction network resulted in a 0.2% increase in mAP@0.5. Overall, the enhancements to the feature extraction block (RVB) and the detection head (LT-Detect) successfully achieved the goal of lightweight design, while the inclusion of EMA and RFB Ensured good detection capabilities.

We conducted a detailed comparison experiment regarding the compression ratio (r) of PConv within the detection head. A comprehensive explanation of the ratio can be found in Section 3.5. We set the compression ratios for the improved detection head's PConv to 3, 4, 5, and 6, and conducted experiments accordingly. The results are as follows (Table 3).

Table 3: Experiments with different values of r

r	P(%)	R(%)	mAP@0.5	Params(M)	FLOPs(G)
None	90.8	81.8	91.6	3.00	3.96
3	88.8	82.8	90.2	2.46	2.71
4	88.5	82.6	90.1	2.42	2.68
5	88.0	80.1	88.8	2.40	2.66
6	87.8	80.0	88.6	2.39	2.65

We observed that setting r to 4 resulted in a noticeable reduction in both parameters and computational load compared to r = 3, without significant loss in detection performance. Conversely, when r = 5, the decrease in parameters and computational load relative to r = 4 was minimal, yet it led to a substantial drop in detection capability. Therefore, we ultimately selected r = 4 for the PConv in our detection head as the optimal choice.

5 Conclusion

This study presents modifications to the YOLOv8n network structure, focusing on enhancing feature extraction, receptive field size, and the detection head. Additionally, a novel Efficient Multi-Scale Attention (EMA) mechanism is incorporated into the feature extraction block. Compared to the original model, these improvements result in approximately 43.3% reduction in parameters and 50% reduction in computational load. Ablation experiments confirm the effectiveness of these enhancements. The revised model achieves a mean Average Precision (mAP@0.5) of 90.9% on a publicly available dataset comprising 3641 images across seven categories of tomato leaf diseases. Furthermore, the improved model maintains strong detection capabilities while significantly reducing its size, effectively addressing the computational limitations encountered in resource-constrained platforms and agricultural machinery application.

Moreover, despite its excellent detection capabilities, this model still has certain limitations. Firstly, when detecting images containing multiple leaves, the model tends to mistakenly identify the healthy portion of a diseased leaf as an independent healthy leaf. Secondly, we observed that, compared to other types of diseases, Yellow Leaf Curl Virus has a higher visual similarity to healthy leaves, resulting in a lower detection rate for this disease. Lastly, this model is specifically designed for detecting tomato leaf diseases, and further research is required to address the detection of tomato leaf pests. This will enable the development of a model capable of simultaneously detecting both diseases and pests in tomato leaves.

Acknowledgement: We express our gratitude to College of Mathematics and Computer Science, Zhejiang Agricultural and Forestry University.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and data collection: Mengjun Tong; analysis, interpretation of results and draft manuscript preparation: Zhenyang He. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The original dataset we used can be found on this website: https://universe.roboflow.com/dyploma/tomato-leaf-diseases-4xa5i (accessed on 10 August 2024).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Costa JM, Heuvelink E. The global tomato industry. In: Tomatoes. UK: CABI Wallingford; 2018. p. 1–26.
2. Vos C, Yang Y, De Coninck B, Cammue B. Fungal (-like) biocontrol organisms in tomato disease control. Biol Control. 2014;74:65–81. doi:10.1016/j.biocontrol.2014.04.004.
3. Singh AK, Ganapathysubramanian B, Sarkar S, Singh A. Deep learning for plant stress phenotyping: trends and future perspectives. Trends Plant Sci. 2018;23(10):883–98. doi:10.1016/j.tplants.2018.07.004.
4. Qi J, Liu X, Liu K, Xu F, Guo H, Tian X. An improved YOLOV5 model based on visual attention mechanism: application to recognition of tomato virus disease. Comput Electron Agric. 2022;194(1):106780. doi:10.1016/j.compag.2022.106780.
5. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–44. doi:10.1038/nature14539.
6. Wang A, Chen H, Lin Z, Han J, Ding G. Repvit: revisiting mobile CNN from VIT perspective. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2024 Jun; Seattle, WA, USA. p. 15909–20.
7. Ouyang D, He S, Zhang G, Luo M, Guo H, Zhan J et al. Efficient multi-scale attention module with cross-spatial learning. In: ICASSP 2023—2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2023; Rhodes, Greece. p. 1–5. doi:10.1109/ICASSP49357.2023.10096516.
8. Chen J, Kao S, He H, Zhuo W, Wen S, Lee CH et al. Run, don't walk: chasing higher flops for faster neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2023 Jun; Vancouver, BC, Canada. p. 12021–31.
9. Liu S, Huang D, Wang A. Receptive field block net for accurate and fast object detection. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018 Sep; Munich, Germany.
10. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2014 Jun; Columbus, OH, USA.
11. Girshick R, Donahue J, Darrell T, Malik J. Region-based convolutional networks for accurate object detection and segmentation. IEEE Trans Pattern Anal Mach Intell. 2016;38(1):142–58. doi:10.1109/TPAMI.2015.2437384.
12. Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans Pattern Anal Mach Intell. 2017;39(6):1137–49. doi:10.1109/TPAMI.2016.2577031.
13. He K, Zhang X, Ren S, Sun J. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans Pattern Anal Mach Intell. 2015;37(9):1904–16. doi:10.1109/TPAMI.2015.2389824.
14. Peng C, Zhao K, Lovell BC. Faster ilod: incremental learning for object detectors based on faster RCNN. Pattern Recognit Lett. 2020;140(2):109–15. doi:10.1016/j.patrec.2020.09.030.
15. Goodfellow IJ, Mirza M, Xiao D, Courville A, Bengio Y. An empirical investigation of catastrophic forgetting in gradient-based neural networks. 2013. doi:10.48550/arXiv.1312.6211.
16. Ren Y, Zhu C, Xiao S. Object detection based on fast/faster RCNN employing fully convolutional architectures. Math Probl Eng. 2018;2018(1):3598316. doi:10.1155/2018/3598316.
17. Sharma VK, Mir RN. Saliency guided faster-RCNN (SGFr-RCNN) model for object detection and recognition. J King Saud Univ-Comput Inf Sci. 2022;34(5):1687–99. doi:10.1016/j.jksuci.2019.09.012.
18. Wang H, Xiao N. Underwater object detection method based on improved faster rcnn. Appl Sci. 2023;13(4):2746. doi:10.3390/app13042746.
19. Redmon J. You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016; Las Vegas, NV, USA.
20. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY et al. SSD: single shot multibox detector. In: Computer Vision—ECCV 2016. Cham: Springer International Publishing; 2016. p. 21–37. doi:10.1007/978-3-319-46448-0_2.
21. Ross T-Y, Dollár G. Focal loss for dense object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017; Honolulu, HI, USA. p. 2980–8.

22. Zhang M, Xu S, Song W, He Q, Wei Q. Lightweight underwater object detection based on yolo v4 and multi-scale attentional feature fusion. Remote Sens. 2021;13(22):4706. doi:10.3390/rs13224706.
23. Cui M, Lou Y, Ge Y, Wang K. LES-YOLO: a lightweight pinecone detection algorithm based on improved YOLOv4-tiny network. Comput Electron Agric. 2023;205:107613. doi:10.1016/j.compag.2023.107613.
24. Li J, Zhu X, Jia R, Liu B, Yu C. Apple-YOLO: a novel mobile terminal detector based on YOLOv5 for early apple leaf diseases. 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC); 2022; Torino, Italy. p. 352–61. doi:10.1109/COMPSAC54236.2022.00056
25. Wang G, Ding H, Yang Z, Li B, Wang Y, Bao L. TRC-YOLO: a real-time detection method for lightweight targets based on mobile devices. IET Comput Vis. 2022;16(2):126–42. doi:10.1049/cvi2.12072.
26. Zhai S, Shang D, Wang S, Dong S. DF-SSD: an improved SSD object detection algorithm based on densenet and feature fusion. IEEE Access. 2020;8:24 344–57. doi:10.1109/ACCESS.2020.2971026.
27. Hu R, Su W-H, Li J-L, Peng Y. Real-time lettuce-weed localization and weed severity classification based on lightweight yolo convolutional neural networks for intelligent intra-row weed control. Comput Electron Agric. 2024;226(10):109404. doi:10.1016/j.compag.2024.109404.
28. Ren X, Bai Y, Liu G, Zhang P. YOLO-Lite: an efficient lightweight network for sar ship detection. Remote Sens. 2023;15(15):3771. doi:10.3390/rs15153771.
29. Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S. End-to-end object detection with transformers. In: Vedaldi A, Bischof H, Brox T, Frahm J-M, editors. Computer Vision—ECCV 2020. Cham: Springer International Publishing; 2020. p. 213–29.
30. Yu W, Si C, Zhou P, Luo M, Zhou Y, Feng J et al. Metaformer baselines for vision. IEEE Trans Pattern Anal Mach Intell. 2024;46(2):896–912. doi:10.1109/TPAMI.2023.3329173.
31. Chu X, Li L, Zhang B. Make RepVGG greater again: a quantization-aware approach. Proc AAAI Conf Artif Intell. 2024 Mar;38(10):11 624–32. doi:10.1609/aaai.v38i10.29045.
32. Hu J, Shen L, Sun G. Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018; Salt Lake City, UT, USA. p. 7132–41.
33. Woo S, Park J, Lee J-Y, Kweon IS. CBAM: convolutional block attention module. In: Proceedings of the European Conference on Computer Vision (ECCV); 2018; Munich, Germany. p. 3–19.
34. Hou Q, Zhou D, Feng J. Coordinate attention for efficient mobile network design. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021; Kuala Lumpur, Malaysia. p. 13713–22.
35. dyploma. Tomato leaf diseases dataset; Apr 2024 [cited 2024 Oct 10]. Available from: https://universe.roboflow.com/dyploma/tomato-leaf-diseases-4xa5i.