




Article

YOLO-FMDI: A Lightweight YOLOv8 Focusing on a Multi-Scale Feature Diffusion Interaction Neck for Tomato Pest and Disease Detection

Hao Sun ^{1,2}, Isack Thomas Nicholas ² , Rui Fu ¹  and Dae-Ki Kang ^{2,*} 

¹ Shandong Facility Horticulture Bioengineering Research Center, Weifang University of Science and Technology, Weifang 262700, China; sunhao@wfust.edu.cn (H.S.); furui19891209@wfust.edu.cn (R.F.)

² Department of Computer Engineering, Dongseo University, 47 Jurye-ro, Sasang-gu, Busan 47011, Republic of Korea; isaactnicholaus@gmail.com

* Correspondence: dkkang@dongseo.ac.kr; Tel.: +82-51-320-1724

Abstract: At the present stage, the field of detecting vegetable pests and diseases is in dire need of the integration of computer vision technologies. However, the deployment of efficient and lightweight object-detection models on edge devices in vegetable cultivation environments is a key issue. To address the limitations of current target-detection models, we propose a novel lightweight object-detection model based on YOLOv8n while maintaining high accuracy. In this paper, (1) we propose a new neck structure, Focus Multi-scale Feature Diffusion Interaction (FMDI), and inject it into the YOLOv8n architecture, which performs multi-scale fusion across hierarchical features and improves the accuracy of pest target detection. (2) We propose a new efficient Multi-core Focused Network (MFN) for extracting features of different scales and capturing local contextual information, which optimizes the processing power of feature information. (3) We incorporate the novel and efficient Universal Inverted Bottleneck (UIB) block to replace the original bottleneck block, which effectively simplifies the structure of the block and achieves the lightweight model. Finally, the performance of YOLO-FMDI is evaluated through a large number of ablation and comparison experiments. Notably, compared with the original YOLOv8n, our model reduces the parameters, GFLOPs, and model size by 18.2%, 6.1%, and 15.9%, respectively, improving the mean average precision (mAP50) by 1.2%. These findings emphasize the excellent performance of our proposed model for tomato pest and disease detection, which provides a lightweight and high-precision solution for vegetable cultivation applications.

Keywords: object detection; deep learning; YOLOv8; FMDI; MFN; UIB; tomato pests and diseases



Citation: Sun, H.; Nicholas, I.T.; Fu, R.; Kang, D.-K. YOLO-FMDI: A Lightweight YOLOv8 Focusing on a Multi-Scale Feature Diffusion Interaction Neck for Tomato Pest and Disease Detection. *Electronics* **2024**, *13*, 2974. <https://doi.org/10.3390/electronics13152974>

Academic Editor: Manohar Das

Received: 2 July 2024

Revised: 20 July 2024

Accepted: 26 July 2024

Published: 28 July 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The increasing prevalence of computer vision technology in agriculture has made the detection of pests and diseases in vegetable crops essential for enhancing agricultural productivity and ensuring crop health. During tomato cultivation, tomatoes are mainly attacked by four pests: leaf blight, leaf curl, septoria leaf spot, and verticillium wilt. Nowadays, tomato cultivation is mostly carried out in greenhouses. The high-humidity and high-temperature environment in greenhouses provides favorable conditions for the survival of pests and diseases. This leads to prolonged harm cycles, making it very challenging to prevent and treat these issues. Consequently, the yield and quality of tomatoes are seriously affected. Traditionally, the detection of these pests and diseases in tomatoes relied on manual inspection, which is a time-consuming and labor-intensive process. In recent years, deep learning-based object-detection models have shown tremendous potential in automating this detection process. Among them, the YOLO (You Only Look Once) family models have attracted much attention due to both their real-time detection ability and high

accuracy [1–5]. The latest iteration, YOLOv8 [6], continues this trend of excellence, further improving efficiency and accuracy across various detection tasks [7].

Despite impressive strides in object-detection technology, implementing these systems on edge devices within vegetable cultivation environments remains a formidable task. The primary hurdles stem from the limited processing power and energy constraints typical of edge computing hardware. This necessitates the development of lightweight, resource-efficient models. While current YOLO iterations demonstrate impressive capabilities, their substantial processing requirements and size make them ill-suited for deployment in resource-limited environments [8–10]. This creates a pressing demand for solutions that balance high detection precision with a streamlined architecture suitable for edge computing. Such advancements are crucial for the swift and accurate detection of pests and diseases, which significantly influence crop productivity and quality [11–13].

Researchers have investigated diverse strategies to enhance the efficiency and precision of object-detection systems. These efforts include refining network structures through novel layer designs and modules [14], as well as exploiting multi-scale feature extraction methods [15,16]. However, such advancements often result in heightened model intricacy and computational requirements. Recent endeavors to boost YOLOv8's capabilities have explored the integration of attention-based mechanisms [17], and feature pyramid architectures [13] and knowledge transfer techniques [18]. Despite these improvements, there remains a need for further optimization to achieve an ideal balance between computational efficiency and detection accuracy. In especially specific applications of detecting pests and diseases such as tomato leaf blight, leaf curl, septoria leaf spot, and verticillium wilt there is a strong need for a real-time target-detection model that is lightweight and highly accurate. Therefore, our study aims to bridge this gap by innovatively modifying the YOLOv8 architecture [19,20].

In this paper, we propose a novel lightweight object-detection model, YOLO-FMDI, based on YOLOv8n, to address the aforementioned challenges. Our contributions are threefold:

- (1) We propose a novel neck structure called the Focus Multi-scale Feature Diffusion Interaction (FMDI), which enables cross-level multi-scale feature diffusion and fusion, significantly enhancing the accuracy of pest and disease detection.

- (2) We propose the Multi-core Focused Network (MFN), an efficient network structure that focuses on capturing context information at different scales and optimizing feature processing capabilities.

- (3) We incorporate the newly proposed Universal Inverted Bottleneck (UIB) [21] module, replacing the original bottleneck module to optimize the C2F network architecture, reducing model complexity and further achieving a lightweight model.

Through extensive ablation and comparative experiments, we validate the superior performance of YOLO-FMDI. Compared with the original YOLOv8n, our model reduces parameters by 18.2%, GFLOPs by 6.1%, and model size by 15.9%, while increasing the mAP50 by 1.2%. We hope that the proposal of YOLO-FMDI will facilitate research in the field of vegetable pest and disease detection.

2. Related Work

2.1. You Only Look Once V8 (YOLOv8)

Among real-time object-detection frameworks, the YOLO series stands out as a pioneer. Its fundamental approach reframes object detection as a regression task, enabling simultaneous prediction of object class and position in a single network traverse [1]. Since its inception, this series has undergone continuous refinement to improve both speed and accuracy, evolving through iterations from YOLOv1 to the current YOLOv8 [2–6]. As depicted in Figure 1A, YOLOv8 incorporates advanced optimization strategies into its architecture, such as the Cross Stage Partial (CSP) [6] framework and an enhanced Feature Pyramid Network (FPN) [6], further enhancing its detection capabilities.

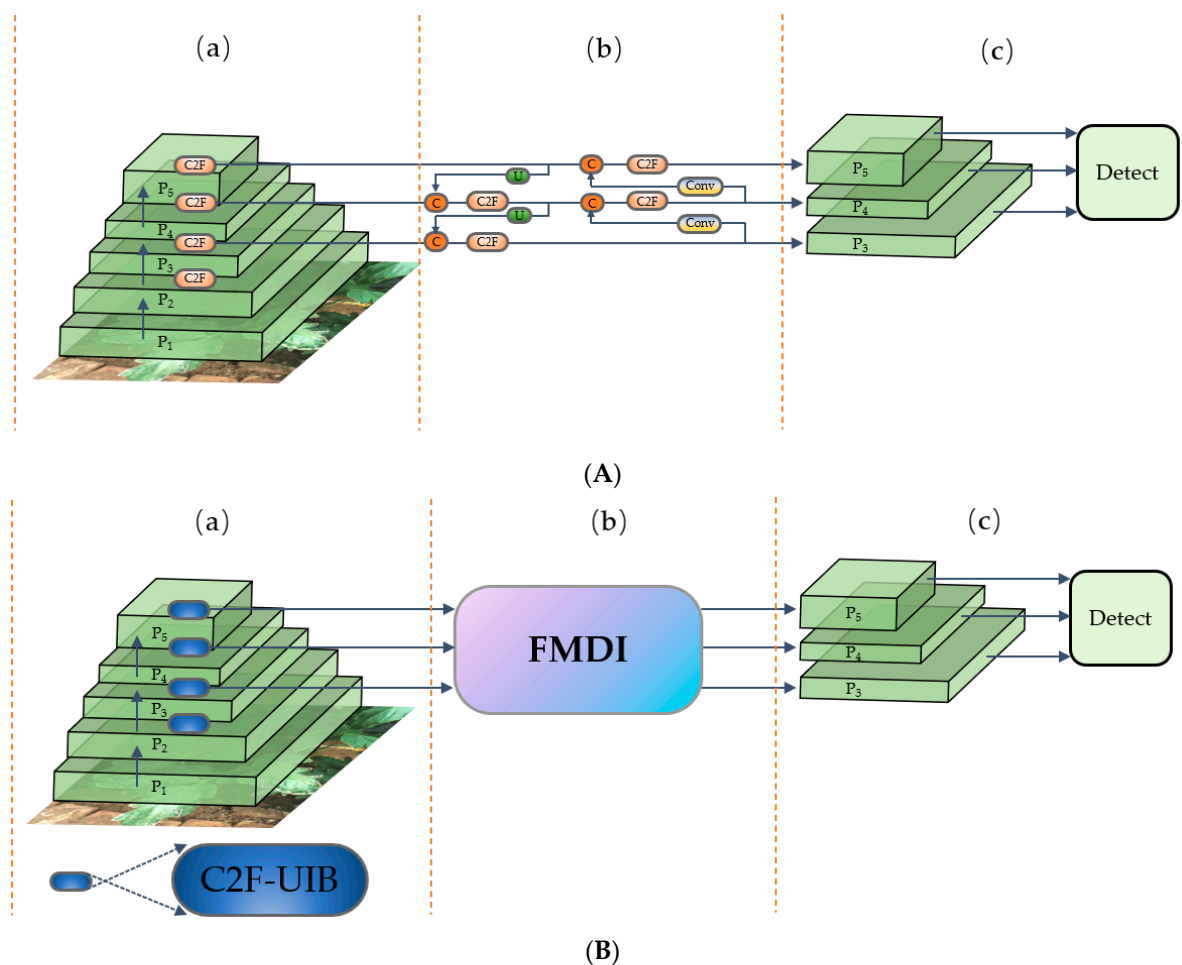


Figure 1. (A) The network architecture of the original YOLOv8 [6]. (B) The overall architecture of YOLO-FMDI. Among them, (a) the model backbone, (b) the model neck part, and (c) the model detection output part.

The pursuit of a streamlined version of the high-performance YOLOv8 base model presents an interesting and complex challenge. By integrating a collection of parallel multi-core deep convolutional layers [8], alongside lightweight UIB [21] and ADown [22] lightweight techniques, we can dramatically reduce the computational requirements of the model while maintaining or even improving its accuracy. This approach provides valuable insights for implementing these models on resource-constrained devices.

2.2. Depthwise Separable Convolution

The deep learning community has embraced depth-wise convolutions [8] as a means to optimize computational efficiency and parameter count without compromising model effectiveness in recent years. Originally introduced in the MobileNet series, these convolutions quickly became a cornerstone of efficient neural network design [8]. Unlike conventional approaches, depth-wise convolutions operate independently on each input channel, substantially reducing computational requirements [8]. This technique has proven particularly effective in architectures like MobileNet [9] and EfficientNet [11], where it is often paired with pointwise (1×1) convolutions to relationships between channels. This depth-wise separable convolution approach has facilitated the development of models that are well suited for use on resource-constrained edge devices. YOLOv8 incorporates multi-scale feature fusion and processing within its neck network to enhance object-recognition capabilities [3]. However, the current implementation relies on standard convolution operations, leading to substantial computational overhead and offering opportunities for

optimization in feature fusion processes. To address this, we propose integrating a set of parallel depth-wise convolutions with diverse kernel sizes and residual connections into YOLOv8's neck network. These parallel branches, each with distinct receptive fields, effectively capture multi-scale features, aiming to boost the feature extraction and integration capabilities of the model and thereby improve the accuracy of detection. Furthermore, since their introduction in ResNet [23], residual connections have been widely recognized for their ability to facilitate gradient flow during training, mitigating the vanishing gradient issue and enabling the effective training of deeper architectures. By combining depth-wise convolutions with residual connections, we aim to further improve feature integration and extraction while maintaining computational efficiency.

2.3. Universal Inverted Bottleneck

The Universal Inverted Bottleneck (UIB) [21] represents a recent innovation in lightweight network architecture, designed to improve efficiency across various computer vision tasks, including object detection and image classification. UIB's key advantage lies in its adaptability, allowing for model optimization without complex search procedures, thus significantly reducing computational requirements. In contrast, conventional bottleneck structures, such as those in ResNet [23] and MobileNetV2 [9], utilize fixed channel expansion ratios, which may prove inefficient when processing different input resolutions or feature dimensions. Subsequent research has built upon and refined this concept. For instance, EfficientNet [11] integrated the inverted bottleneck with additional optimization techniques through compound scaling, achieving a balanced performance across multiple resource constraints. UIB's design preserves the benefits of the inverted bottleneck while increasing the network's expressive capacity and computational efficiency. This is achieved by sharing common components like pointwise expansion and projection, while treating only the depth-wise convolution as a variable element. YOLOv8 incorporates the Cross Stage Partial connections with the full-stage fusion (C2F) [6] module to bolster its feature extraction and fusion capabilities. However, the bottleneck components within C2F can incur significant computational costs under certain conditions. Given UIB's adaptive nature, which is particularly advantageous in overcoming the challenges of varying feature dimensions within the C2F module, its integration offers the potential to markedly reduce the model's computational complexity and parameter count. The aim of this approach is to maintain detection accuracy while achieving a more lightweight YOLOv8 model.

3. Methodology

3.1. Overall Architecture of the YOLO-FMDI

The overall architecture of our newly proposed YOLO-FMDI is illustrated in Figure 1B and can be summarized into three main parts: (a) the backbone with the improved C2F with UIB; (b) the neck with a focus on the Multi-scale Feature Diffusion Interaction module; (c) the original detection part. First, a 640×640 input image is fed into P1, where it undergoes a series of convolutions and C2F modules, resulting in features with $1/8$, $1/16$, and $1/32$ resolutions at P3, P4, and P5, respectively. Subsequently, the feature maps with different dimensions obtained from the backbone are inputted into the FMDI structure, which undergoes N stages of feature focusing, feature diffusion, and feature interaction enhancement, thereby acquiring multi-scale features with richer semantic information. Finally, the feature maps after N stages of feature interaction are passed into the detection part for pest and disease recognition tasks.

3.2. Focus Multi-Scale Feature Diffusion Interaction

We propose a module called FMDI, standing for Focus on Multi-scale Feature Diffusion Interaction, as illustrated in Figure 2. It takes three scaled features from backbone modules as inputs to the Multi-core Focused Network (MFN), focusing on multi-scale features, allowing each input scale of these features to possess detailed contextual semantic information, further enhancing the network's modeling and semantic representation capabilities.

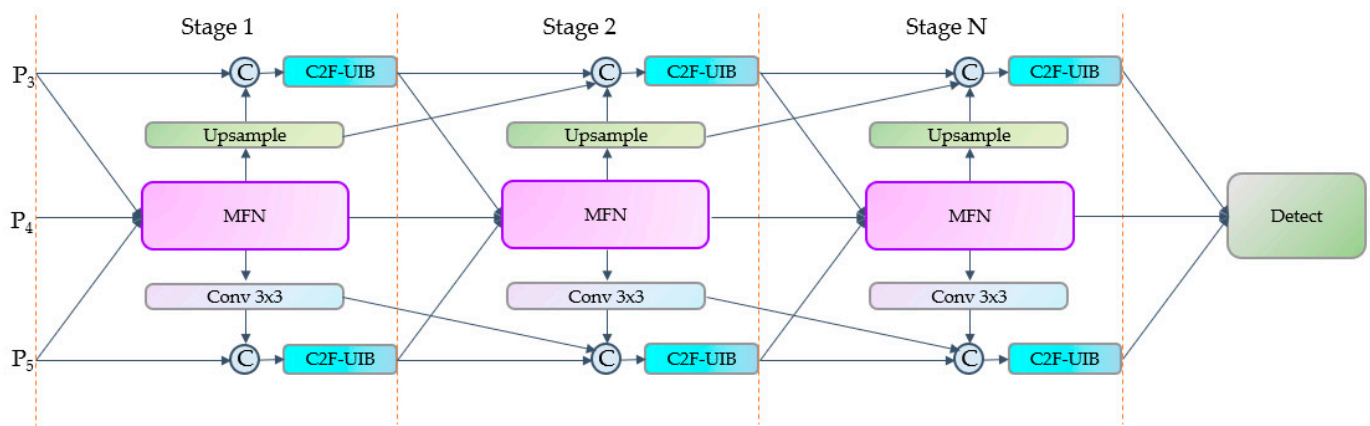


Figure 2. Focus Multi-scale Feature Diffusion Interaction module.

Subsequently, the features focused by the MFN are processed through three branches: (1) by upsampling and fusing with the P3 scale features, then passing through a lightweight C2F-UIB module, mixing semantic information from different spatial and channel dimensions; (2) by a 3×3 convolutional downsampling operation and fusing with the P5 scale features, also passing through another lightweight C2F-UIB module; (3) directly outputting from the MFN. Finally, the multi-scale features outputted from the above three branches serve as the input for the MFN module with three different scales of features in the next stage. Simultaneously, starting from the second stage, the upsampling and downsampling features from the previous stage are also inputted to the next stage, as shown at the Concat position in Figure 2. This process is repeated until the Nth stage is completed. Through this focusing and diffusion mechanism, features with rich contextual semantic information are diffused to various detection scales, further enhancing the expressiveness and generalization ability of the model.

3.3. Multi-Core Focused Network

The Multi-core Focused Network (MFN) consists of multi-scale pyramid layers and multi-core deep convolutional layers. In the multi-scale pyramid layers, upsampling, Conv 1×1 , and the novel lightweight ADown [22] downsampling are introduced. The ADown [22] module reduces model complexity by optimizing the number of parameters in the convolutional layers while preserving as much image information as possible without affecting object-detection accuracy. Meanwhile, the multi-core deep convolutional layers extend the receptive field through different convolutional kernel sizes to capture multi-scale texture features, thereby enhancing the network's long-range modeling capability, as shown in Figure 3. Specifically, after focusing and fusing the features, a set of parallel depth-wise convolutional layers with different receptive fields and residual structures are introduced to capture cross-scale contextual semantic information (e.g., $k = 5 \times 5, 7 \times 7, 9 \times 9, 11 \times 11$). Subsequently, a Conv 1×1 and residual structure are used to further capture more local information. This process can be represented as:

$$C(P) = C(AD(P3), \text{Conv}(P4), \text{Conv}(\text{Up}(P5))) \quad (1)$$

$$O = C(P) + \text{Conv}(\text{DWConv}_{k \times k}(C(P))) + C(P) \quad (2)$$

where $C(\cdot)$ is Concat, $AD(\cdot)$ is ADown, $\text{Up}(\cdot)$ is Upsample, $\text{Conv}(\cdot)$ is Conv 1×1 , and $\text{DWConv}_{k \times k}(\cdot)$ is a set of depth-wise convolutions with different kernel sizes.

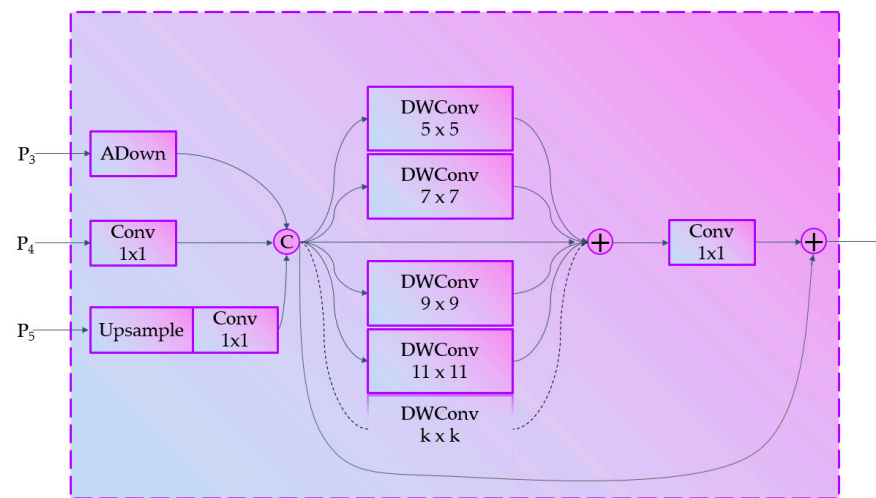


Figure 3. Multi-core Focused Network.

3.4. Improved C2F with UIB

The C2F module in YOLOv8 has evolved from the C3 module in YOLOv5. As shown in Figure 4, within the bottleneck block, the input first passes through a 1×1 convolutional layer, followed by a 3×3 convolutional layer, and is finally added to the initial value through a residual structure. As can be seen, the C2F block contains more skipped connections, eliminates the convolution operation in branches, and introduces an additional splitting operation. This structural design allows for the capture of richer feature information. To further lightweight this module while maintaining detection accuracy, we incorporate the new and efficient Universal Inverted Bottleneck (UIB) block to replace the original bottleneck block. This module can flexibly adapt to building models with various optimization objectives without the need for complex searches. While maintaining a similarly simple structure, it shares common modules such as pointwise expansion and projection, enabling the C2F module to temporarily make trade-offs between spatial and channel mixing. The optimization of the above modules further maximizes computational utilization and serves to lightweight the model.

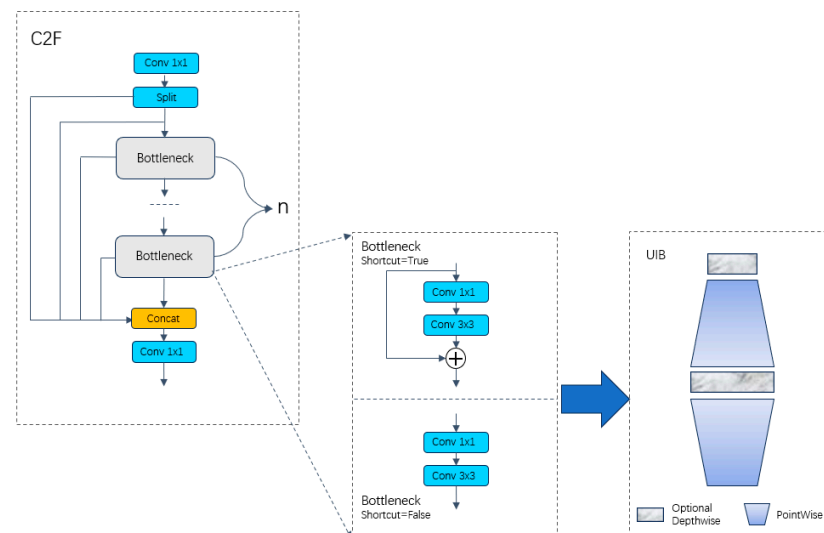


Figure 4. C2F-UIB module.

4. Experiment

The experimental dataset comprised 1810 images of healthy and spotted tomato leaves collected by us, along with 2700 images of leaf curl, wilt, and blight diseases [24]. In our

experiments, we divided the 4510 images in the tomato pest and disease dataset according to the ratio of 8:1:1, specifically categorized as a training set of 3608 images, a validation set of 451 images, and a test set of 451 images. Tomato leaf blight, leaf curl, septoria leaf spot, and verticillium wilt are important phytopathological concerns in tomato cultivation. The fungus *Alternaria solani* induces leaf blight, which is characterized by concentric, necrotic brown lesions that result in early leaf shedding and diminished photosynthetic efficiency. Various begomoviruses are responsible for leaf curl, which causes leaves to yellow and curl upwards, inhibiting plant development and reducing fruit production [25]. *Septoria lycopersici* is the pathogen behind septoria leaf spot, manifesting as small, round lesions featuring dark borders and pale centers, which merge over time, inflicting widespread foliar harm [25]. Verticillium wilt, caused by either *Verticillium dahliae* or *V. albo-atrum*, is identifiable by the wilting, yellowing, and death of lower foliage, often affecting one side of the plant due to vascular system invasion [26]. These pathogens severely disrupt crucial physiological functions in tomato plants, including carbon fixation, nutrient circulation, and water absorption [27]. The consequences include decreased plant vitality, restricted growth, and accelerated aging. Collectively, these diseases lead to significant reductions in crop yield, deterioration of fruit quality, and lowered market value, presenting substantial financial obstacles for tomato growers globally. Due to the limitations imposed by the local cultivation conditions at the time, we could not obtain physical samples of leaf curl, wilt, and blight diseases. Consequently, we supplemented our dataset with the images of these three disease categories from [24]. Partial dataset images are displayed in Figure 5. We conducted extensive ablation and comparative experiments on this curated dataset to validate the efficacy of our proposed optimized model in detecting pests and diseases affecting vegetation.

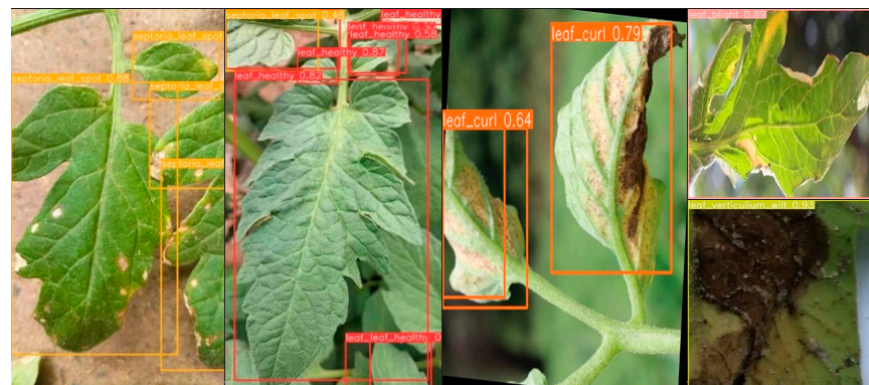


Figure 5. Selected sample plots of the dataset.

4.1. Experimental Setups

The experiments were conducted on a platform built on the Windows 11 (64-bit) operating system. The specific environmental parameters used are presented in Tables 1 and 2.

Table 1. Experimental configuration environment.

Platform Environment	Details
Operating system	Windows 11 (64-bit)
Programming language	Python 3.8.19
Memory	32G
CPU	AMD Ryzen 9 7950X3D 16-Core Processor 4.20 GHz
GPU	NVIDIA GeForce RTX 4090
CUDA	11.8
Pytorch	2.2.2
Development platform	Visual Studio Code

Table 2. Experimental parameter settings.

Parameter	Value
Initial learning rate	0.01
Final learning rate	0.01
Momentum	0.937
Batch size	12
Input image size	640×640
Epoch	130

4.2. Evaluation Indicators

In this paper, we utilized the model parameter counts, GFLOPs, and size of the model as quantitative metrics for evaluating the degree to which the lightweight model has been achieved. Concurrently, we utilized the mean average precision (mAP50 and mAP50-95) to assess the accuracy of the model. The formulation for computing precision (P) is delineated in Equation (2), recall (R) in Equation (3), and the mean average precision (mAP) in Equation (4).

$$P = \frac{TP}{TP + FP} \quad (3)$$

$$R = \frac{TP}{TP + FN} \quad (4)$$

$$AP = \int_0^1 P(R) dR \quad (5)$$

$$mAP = \frac{\sum_{i=1}^K AP_i}{K} \quad (6)$$

In Equations (3) to (6), we used several key metrics to evaluate the model performance. Among them, TP represents the number of accurately identified tomato pest instances, FP represents the number of healthy instances misclassified as pests, and FN refers to the number of actual pest instances that were not successfully detected. Precision (P) and recall (R) are two important evaluation metrics, and their different combinations form a curve. AP value, which is the area under this curve, reflects the overall performance of the model at different thresholds. The mAP, on the other hand, is the average of multiple AP values and is used to measure the overall performance of the model across multiple categories.

4.3. Ablation Studies

4.3.1. Ablation for Components

In the ablation experiments, we progressively incorporated the respective submodules into the YOLOv8n baseline model, ultimately evolving into our proposed optimized model, YOLO-FMDI. The results of the ablation experiments are presented in Table 3. When the original neck structure is replaced with the FMDI structure incorporating the original C2F module, an improvement of 1.1% in mAP50 and 0.7% in mAP50-95 is observed. However, using the FMDI structure with the C2F module fused with UIB yields an improvement of 1.8% in mAP50 and 3.2% in mAP50-95 is observed. As the objective of this paper is to produce a lightweight model, the two ablation approaches mentioned above resulted in an increase in either parameters, GFLOPs, or model size. Considering the lightweight capabilities of the UIB module, we replaced solely the bottleneck blocks within all C2F modules with UIB, leaving the remaining modules unaltered. This resulted in a 27.5% reduction in the number of parameters, a 25.6% decrease in GFLOPs, and a 25.4% reduction in model size. Ultimately, we constructed our proposed optimized model by replacing all bottleneck blocks within the C2F modules with UIB, while simultaneously substituting the original neck structure of the YOLOv8 model with FMDI. In summary, compared with the standard YOLOv8 model, our proposed model achieved an improvement of 1.2% in mAP50 and 0.9% in mAP50-95, along with reductions of 18.2% in parameters, 6.1% in GFLOPs,

and 15.9% in model size. The experimental results demonstrate that our proposed FMDI structure and UIB block fusion can significantly optimize the capabilities of the standard YOLOv8, and better adapt it to the task of tomato pest and disease detection.

Table 3. Ablation studies of key components.

Model	Parameters	GFLOPs	Model Size	mAP50	mAP50-95
YOLOv8n(baseline)	3,011,823	8.2	6.3 MB	0.928	0.745
+FMDI(C2F)	3,048,559	9.4	6.4 MB	0.938	0.750
+FMDI(UIB)	2,884,575	8.9	6.1 MB	0.945	0.769
+UIB	2,184,031	6.1	4.7 MB	0.917	0.728
+UIB+ FMDI(UIB) (ours)	2,463,519	7.7	5.3 MB	0.939	0.752

4.3.2. Number of Focus Diffusion Interactions

In Table 4, we list the impact of varying the number of focus diffusion interaction stages on the model's characteristics. We observed that as the stage count N increases, the model's parameters, GFLOPs, and model size metrics exhibit a continuous rise. However, at $N = 2$, the model attains the highest mAP50 value, while concurrently maintaining its lightweight metrics lower than the YOLOv8n baseline model. Consequently, we set the default stage N value to 2.

Table 4. Ablation of the number of focus diffusion interactions. The model performs best when $N = 2$.

The number of Stages (N)	Parameters	GFLOPs	Model Size	mAP50	mAP50-95
1	1,864,927	6.0	4.0 MB	0.915	0.738
2 (ours)	2,463,519	7.7	5.3 MB	0.939	0.752
3	3,396,863	10.6	7.2 MB	0.927	0.747
4	4,927,375	15.2	10.3 MB	0.931	0.758

4.3.3. Different Kernel Sizes in MFN

Table 5 illustrates the impact of varying kernel sizes on the Multi-core Focused Network (MFN). The results demonstrate that as the kernel size increases, there is a concomitant escalation in the model's parameters, GFLOPs, and model size metrics. Simultaneously, we observed that the utilization of kernel sizes 5, 7, 9, and 11 yields the apex values for mAP50 and mAP50-95. Consequently, we ultimately determined to employ 5, 7, 9, and 11 as the default kernel size configuration for the MFN.

Table 5. Ablation of the setting of kernel size in MFN.

Kernel Size (k)	Parameters	GFLOPs	Model Size	mAP50	mAP50-95
3	2,333,919	7.3	5.0 MB	0.905	0.722
3,5	2,346,399	7.4	5.0 MB	0.924	0.740
3,5,7	2,370,399	7.5	5.1 MB	0.931	0.748
3,5,7,9	2,409,759	7.6	5.1 MB	0.925	0.746
3,5,7,9,11	2,468,319	7.8	5.3 MB	0.929	0.754
3,5,7,9,11,13	2,549,919	8.0	5.4 MB	0.923	0.740
5	2,341,599	7.4	5.0 MB	0.920	0.744
5,7	2,365,599	7.4	5.1 MB	0.925	0.743
5,7,9	2,404,959	7.6	5.1 MB	0.916	0.730
5,7,9,11 (ours)	2,463,519	7.7	5.3 MB	0.939	0.752
5,7,9,11,13	2,545,119	8.0	5.4 MB	0.921	0.739

4.3.4. Comparison Studies

Table 6 show the comparative analysis of the YOLO-FMDI model against various object-detection models. It is evident that our model exhibits superior lightweight metrics in terms of parameters, GFLOPs, and model size when compared with other network models.

For instance, our model boasts the lowest parameters at 2,463,519, the lowest GFLOPs at 7.7, and the most diminutive model size at 5.3 MB. Furthermore, while maintaining its lightweight advantage, our model demonstrates outstanding performance in terms of mean average precision, with commendable mAP50 and mAP50-95 values. Consequently, our model is better suited for the tomato pest and disease detection task at hand.

Table 6. Object detection with different frameworks.

Model	Parameters	GFLOPs	Model Size	FPS	mAP50	mAP50-95
YOLOv5s [5]	46,563,709	109.9	27.0 MB	92.9	0.553	0.364
YOLOx [28]	54,208,895	156.0	34.4 MB	76.6	0.853	0.599
YOLO7 [29]	3,762,012	106.5	74.3 MB	64.5	0.847	0.638
DETR [30]	3,674,045	74.0	15.9 MB	42.6	0.507	0.395
YOLO7-tiny [29]	6,529,483	13.9	23.2 MB	108.5	0.618	0.404
YOLOv8n [6] (baseline)	3,011,823	8.2	6.3 MB	228.1	0.928	0.745
YOLO-FMDI (ours)	2,463,519	7.7	5.3 MB	311.4	0.939	0.752

As shown in Figure 6 the visualization of the detection results of the three different models is displayed. We selected the models with the clearest difference in detection results, YOLOv5s and YOLO-FMDI, and also selected the baseline model YOLOv8n. From the detection results of diseased tomato leaves, the mean average precision (mAP50) of YOLOv5s, YOLOv8n, and YOLO-FMDI in detecting leaf verticillium wilt were 0.54, 0.89 and 0.92, respectively. The mAP50 of our proposed model is 0.03 higher than the mAP50 of the baseline model and 0.38 higher than the mAP50 of the YOLOv5s model. Meanwhile, it can be seen that the mAP50 of the YOLO-FMDI model in detecting leaf curl is 0.91, which is 0.08 higher than the 0.83 of the YOLOv8n model and 0.30 higher than the 0.61 of the YOLOv5s model. Therefore, the detection ability of our proposed model YOLO-FMDI is better than the other models.

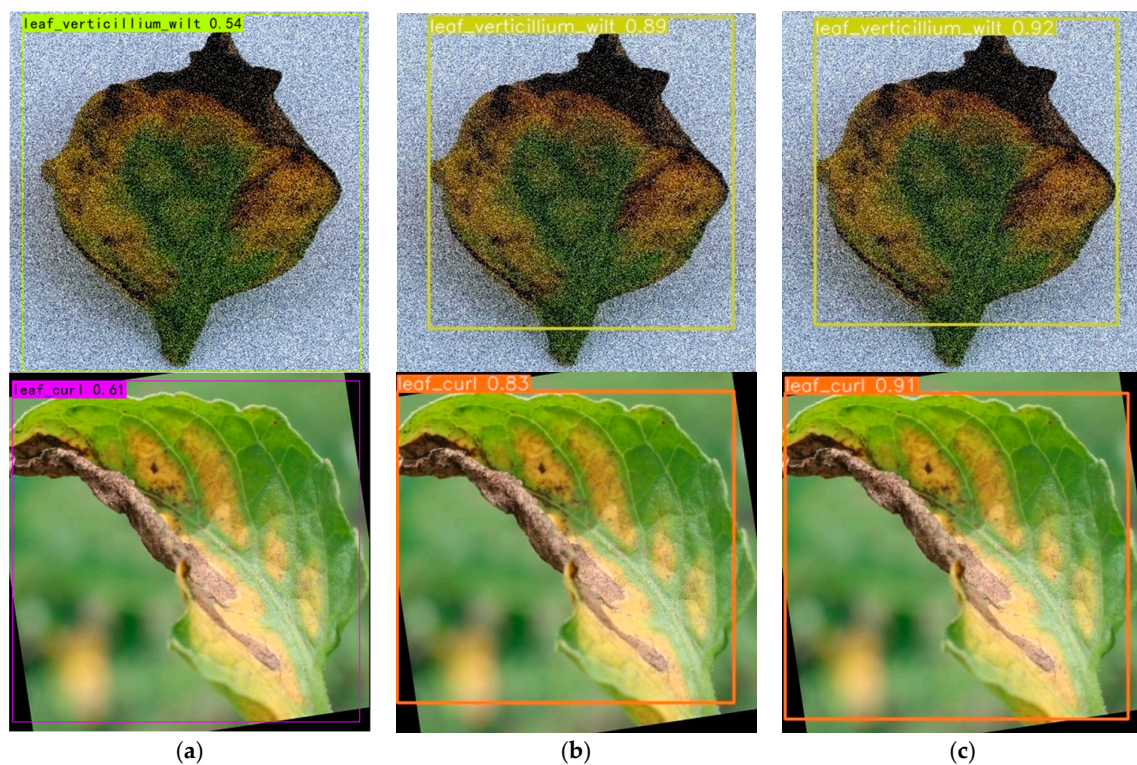


Figure 6. Comparison of the visualization detection result of different models, (a) YOLOv5s; (b) YOLOv8n; (c) YOLO-FMDI.

5. Conclusions

In this work, we propose the YOLO-FMDI model, which incorporates a novel Focus Multi-scale Feature Diffusion Interaction neck structure. It effectively diffuses and fuses focused multi-scale features, reconstructing fine-grained hierarchical semantic information. Simultaneously, we integrate the state-of-the-art lightweight Universal Inverted Bottleneck module. We validate the effectiveness of YOLO-FMDI on a tomato pest and disease dataset. Extensive ablation and comparative experiments demonstrate that our approach achieves higher detection accuracy while attaining lightweighting compared to conventional object-detection models. Moving forward, we aim to collect a more diverse sample dataset from local vegetable cultivation environments and continue to lightweight advanced real-time monitoring models, providing robust technical support for vegetable cultivation.

Author Contributions: Conceptualization, H.S. and D.-K.K.; Data curation, H.S. and R.F.; Formal analysis, H.S., R.F., I.T.N. and D.-K.K.; Funding acquisition, D.-K.K.; Investigation, I.T.N.; Methodology, H.S. and R.F.; Project administration, H.S. and D.-K.K.; Software, H.S.; Supervision, I.T.N. and D.-K.K.; Validation, H.S. and D.-K.K.; Visualization, H.S.; Writing—original draft, H.S.; Writing—review and editing, D.-K.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2022R1A2C2012243).

Data Availability Statement: This study analyzed a combination of publicly available datasets and data collected by the authors. The publicly available datasets can be accessed at [<https://data.mendeley.com/datasets/bwh3zbpkpv/1>] (accessed on 2 July 2024). However, due to the sensitive nature of the research, the dataset collected by the authors is not publicly available.

Acknowledgments: The authors wish to thank the members of the Dongseo University Machine Learning/Deep Learning Research Lab, and the anonymous referees for their helpful comments on earlier drafts of this paper.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
2. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
3. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**. [[CrossRef](#)]
4. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**. [[CrossRef](#)]
5. Home—Ultralytics YOLOv5 Docs. Available online: <https://docs.ultralytics.com/zh/models/yolov5/> (accessed on 2 July 2024).
6. Home—Ultralytics YOLOv8 Docs. Available online: <https://docs.ultralytics.com/> (accessed on 2 July 2024).
7. Wang, C.-Y.; Liao, H.-Y.M.; Wu, Y.-H.; Chen, P.-Y.; Hsieh, J.-W.; Yeh, I.-H. CSPNet: A New Backbone That Can Enhance Learning Capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 390–391.
8. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**. [[CrossRef](#)]
9. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. *arXiv* **2019**. [[CrossRef](#)]
10. Howard, A.; Sandler, M.; Chu, G.; Chen, L.-C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for MobileNetV3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27–28 October 2019; pp. 1314–1324.
11. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
12. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 10781–10790.
13. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37.

14. Lin, T.-Y.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
15. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as Points. *arXiv* **2019**. [[CrossRef](#)]
16. Law, H.; Deng, J. CornerNet: Detecting Objects as Paired Keypoints. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 734–750.
17. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
18. Chen, G.; Choi, W.; Yu, X.; Han, T.; Chandraker, M. Learning Efficient Object Detection Models with Knowledge Distillation. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Curran Associates, Inc.: Dutchess County, NY, USA, 2017; Volume 30.
19. Zhao, Z.-Q.; Zheng, P.; Xu, S.-T.; Wu, X. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [[CrossRef](#)] [[PubMed](#)]
20. Duan, K.; Bai, S.; Xie, L.; Qi, H.; Huang, Q.; Tian, Q. CenterNet: Keypoint Triplets for Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27–28 October 2019; pp. 6569–6578.
21. Qin, D.; Leichner, C.; Delakis, M.; Fornoni, M.; Luo, S.; Yang, F.; Wang, W.; Banbury, C.; Ye, C.; Akin, B.; et al. MobileNetV4—Universal Models for the Mobile Ecosystem. *arXiv* **2024**, arXiv:2404.10518. Available online: <http://arxiv.org/abs/2404.10518> (accessed on 26 July 2024).
22. Wang, C.-Y.; Yeh, I.-H.; Liao, H.-Y.M. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. *arXiv* **2024**, arXiv:2402.13616.
23. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
24. Mensah, P.K.; Akoto-Adjepong, V.; Adu, K.; Ayidzoe, M.A.; Bediako, E.A.; Nyarko-Boateng, O.; Boateng, S.; Donkor, E.F.; Bawah, F.U.; Awarayi, N.S.; et al. CCMT: Dataset for Crop Pest and Disease Detection. *Data Brief* **2023**, *49*, 109306. [[CrossRef](#)] [[PubMed](#)]
25. Panthee, D.R.; Chen, F. Genomics of fungal disease resistance in tomato. *Hortic. Res.* **2021**, *8*, 30–39. [[CrossRef](#)] [[PubMed](#)]
26. Akhtar, K.P.; Saleem, M.Y.; Asghar, M.; Haq, M.A. Resistance of Solanum species to Cucumber mosaic virus sub group IA and its vector Myzus persicae. *Eur. J. Plant Pathol.* **2019**, *153*, 115–125.
27. Fuentes, A.; Yoon, S.; Kim, S.C.; Park, D.S. A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. *Sensors* **2017**, *17*, 2022. [[CrossRef](#)] [[PubMed](#)]
28. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. YOLOX: Exceeding YOLO Series in 2021. *arXiv* **2021**. [[CrossRef](#)]
29. Wang, C.-Y.; Bochkovskiy, A.; Liao, H.-Y.M. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. *arXiv* **2022**. [[CrossRef](#)]
30. Dai, X.; Chen, Y.; Yang, J.; Zhang, P.; Yuan, L.; Zhang, L. Dynamic DETR: End-to-End Object Detection with Dynamic Attention. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 2988–2997.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.