sustainability

MDPI

Article

A CNNA-Based Lightweight Multi-Scale Tomato Pest and Disease Classification Method

Yanlei Xu [1], Zhiyuan Gao [1], Yuting Zhai [1], Qi Wang [1], Zongmei Gao [2], Zhao Xu [3] and Yang Zhou [1,*]

[1] College of Information and Technology, JiLin Agricultural University, Changchun 130118, China; yanleixu@jlau.edu.cn (Y.X.); gzy@mails.jlau.edu.cn (Z.G.); 20221850@mails.jlau.edu.cn (Y.Z.); 20211381@mails.jlau.edu.cn (Q.W.)
[2] Center for Precision and Automated Agricultural Systems, Department of Biological Systems Engineering, Washington State University, Prosser, WA 99350, USA; zongmei.gao@wsu.edu
[3] Non Commissioned Officer School, Army Academy of Armored Force, Changchun 130118, China; xuzhaogongzuo@163.com
* Correspondence: zhouyang@jlau.edu.cn

Abstract: Tomato is generally cultivated by transplanting seedlings in ridges and furrows. During growth, there are various types of tomato pests and diseases, making it challenging to identify them simultaneously. To address this issue, conventional convolutional neural networks have been investigated, but they have a large number of parameters and are time-consuming. In this paper, we proposed a lightweight multi-scale tomato pest and disease classification network, called CNNA. Firstly, we constructed a dataset of tomato diseases and pests consisting of 27,193 images with 18 categories. Then, we compressed and optimized the ConvNeXt-Tiny network structure to maintain accuracy while significantly reducing the number of parameters. In addition, we proposed a multi-scale feature fusion module to improve the feature extraction ability of the model for different spot sizes and pests, and we proposed a global channel attention mechanism to enhance the sensitivity of the network model to spot and pest features. Finally, the model was trained and deployed to the Jetson TX2 NX for inference of tomato pests and diseases in video stream data. The experimental results showed that the proposed CNNA model outperformed the pre-trained lightweight models such as MobileNetV3, MobileVit, and ShuffleNetV2 in terms of accuracy and all parameters, with a recognition accuracy of 98.96%. Meanwhile, the error rate, inference time for a single image, network parameters, FLOPs, and model size were only 1%, 47.35 ms, 0.37 M, 237.61 M, and 1.47 MB, respectively.

Keywords: tomato; plant disease; pest; convolutional neural network; deep learning; lightweight; multi-scale; attention mechanism

check for updates

Citation: Xu, Y.; Gao, Z.; Zhai, Y.; Wang, Q.; Gao, Z.; Xu, Z.; Zhou, Y. A CNNA-Based Lightweight Multi-Scale Tomato Pest and Disease Classification Method. *Sustainability* **2023**, *15*, 8813. https://doi.org/10.3390/su15118813

Academic Editors: Rocío Calderón Madrid, Carlos Camino and Tomás Poblete

Received: 18 April 2023
Revised: 19 May 2023
Accepted: 20 May 2023
Published: 30 May 2023

Copyright: © 2023 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

Tomatoes are highly sought after worldwide as both a fruit and vegetable. However, the growth process of tomato plants is often plagued by a variety of pests and diseases, which seriously hinder the development of the tomato growing industry and affect farmers. Effective and accurate identification of tomato pests and diseases is an important tool for their management and an important prerequisite for improving tomato production [1,2]. Traditional tomato disease identification is typically carried out exclusively by experts or technicians due to their high accuracy, but the methods are not widely applicable due to their time-consuming, costly, and inefficient nature [3].

Conventional machine learning-based methods have been widely employed in research exploring the intelligent identification of plant diseases [4–6]. Mokhtar et al. [7] utilized a support vector machine (SVM) algorithm with different kernel functions for the classification and identification of tomato mosaic disease with an average accuracy of 92%. Johannes et al. [8] identified three diseases of wheat leaves using the Naïve Bayes technique

with an accuracy of 85%. Rumpf et al. [9] used SVM to detect three diseases in sugar beet root images with an accuracy of 86%. Although these machine learning algorithms can generally satisfy the requirements of disease identification, feature extraction is a complex process that can reduce recognition accuracy and efficiency.

With the development of deep learning, convolutional neural network (CNN) models have been developed to autonomously extract image features and perform classification with higher accuracy and efficiency [10]. Yang et al. [11] proposed a fine-grained classification model, LFC-Net, with a self-supervised mechanism to classify images of eight tomato diseases and healthy leaf images with 99.7% accuracy. Ji et al. [12] proposed a joint model architecture based on an integrated approach to classify four grape leaf diseases in the open dataset of PlantVillage with 98.57% accuracy. Anandhakrishnan et al. [13] proposed a deep convolutional neural network model to classify tomato leaf disease using the open dataset in PlantVillage with 98.40% accuracy. Despite their high accuracy in plant disease identification, these convolutional neural networks had certain limitations, such as a large number of network parameters and slow model inference, which required attention.

Therefore, researchers have started to apply lightweight modeling algorithms to disease identification [14–17]. Elhassouny et al. [18] used the lightweight network model MobileNet to identify 10 common tomato leaf diseases and compared the results of several different optimizers to achieve a final accuracy of 90.3%. Agarwal et al. [19] proposed a convolutional neural network model with only three convolutional layers and three maximum pooling layers and a convolutional neural network model consisting of two fully connected layers to classify tomato disease using the PlantVillage open dataset with an accuracy of 91.2%. Wang et al. [20] proposed a shallow network with two to ten convolutional layers to classify four apple diseases using the PlantVillage open dataset and obtained an accuracy of 79.3%. Hamid, et al. [21] used MobileNetV2 for classification of 14 different categories of seeds and achieved an accuracy of 95% on the test sets. While these lightweight models featured short training times and fast inference speeds, their recognition accuracy was relatively low.

To further improve model recognition accuracy, attention mechanisms focusing on improving the model's attention to key information have been proposed, with common attention mechanisms such as squeeze-and-excitation (SE) [22], convolutional block attention module (CBAM) [23], etc. Yin et al. [24] proposed a deep learning network, DISE-Net, introducing a coordinate attention mechanism to construct and classify a field corn leaflet spot dataset with an accuracy of 97.12%, which was 2.21% higher than the model without introducing the attention mechanism. Gao et al. [25] proposed a dual-branch, efficient, channel attention mechanism-based DECA_ResNet model for cucumber disease recognition. The model was trained on the Global AI Challenge 2018 dataset, the PlantVillage dataset, and a self-collected cucumber disease dataset. The model accuracy reached 98.54%, which was 7.66% higher than the recognition accuracy without introducing the attention mechanism. The introduction of attention mechanisms has been able to improve the accuracy of models in identifying diseases. In addition, in current research work, most models are limited to plant diseases, and there are very few studies on the simultaneous identification of plant diseases and insect pests. In fact, insect pests also hinder the growth of plants, so a network model for the simultaneous identification of diseases and insect pests is needed. The introduction of the above attention mechanisms can improve the sensitivity of the model in the disease region, whereas the model is less sensitive in the insect pest region. Therefore, improving the attention mechanism to simultaneously increase the sensitivity of the model to both diseases and pests is urgently needed.

In summary, current research efforts to identify tomato pests and diseases have the following challenges:

(1) In terms of content recognition, it is still difficult to collect pest dataset in the field and obtain high model performance for simultaneous recognition of pests and diseases. Generally, researchers are currently focused on disease identification without achieving unified identification of diseases, insect pests, and healthy leaves.

(2) In terms of model performance, most studies achieve single performance improvement in terms of model accuracy, size, or robustness, without achieving a comprehensive balance of these performances.

To address these challenges, our study makes the following contributions:

(1) We built a dataset consisting of 27,193 images across 18 categories, including tomato diseases, pests, and healthy leaves.
(2) We proposed an efficient and lightweight classification model named ConvNeXt-Nano-Adjust (CNNA), which accurately and rapidly classified images of tomato diseases and pests.
(3) We embedded the CNNA model into the Jetson TX2 NX with an inference time of only 47.35 ms per image, which is suitable for practical production applications. This approach can provide technical support for the development of a management and control system for tomato diseases and pests.

2. Materials and Methods

The workflow of this study is shown in Figure 1. Firstly, we constructed a dataset of images of tomato diseases and insect pests, as shown in Figure 1a. The tomato disease dataset was obtained from the PlantVillage open dataset, whereas the tomato pest leaf images were collected from tomato plants in an experimental field. Image preprocessing was performed on the original images. Images of leaves with diseases, pests, and in healthy condition were combined and divided into the training and validation sets. Secondly, the ConvNeXt-Tiny network was developed and optimized, and the lightweight multi-scale feature fusion module (MFFM) and lightweight global channel attention (GCA) mechanism were proposed to further optimize the model, as shown in Figure 1b. Specifically, we conducted horizontal and vertical performance comparison experiments for optimizing the model; the model weight file is shown in Figure 1d. Finally, the model with the best validation accuracy was deployed to the Jetson TX2 NX for inference of tomato leaf diseases and pests in video stream data, as shown in Figure 1e.

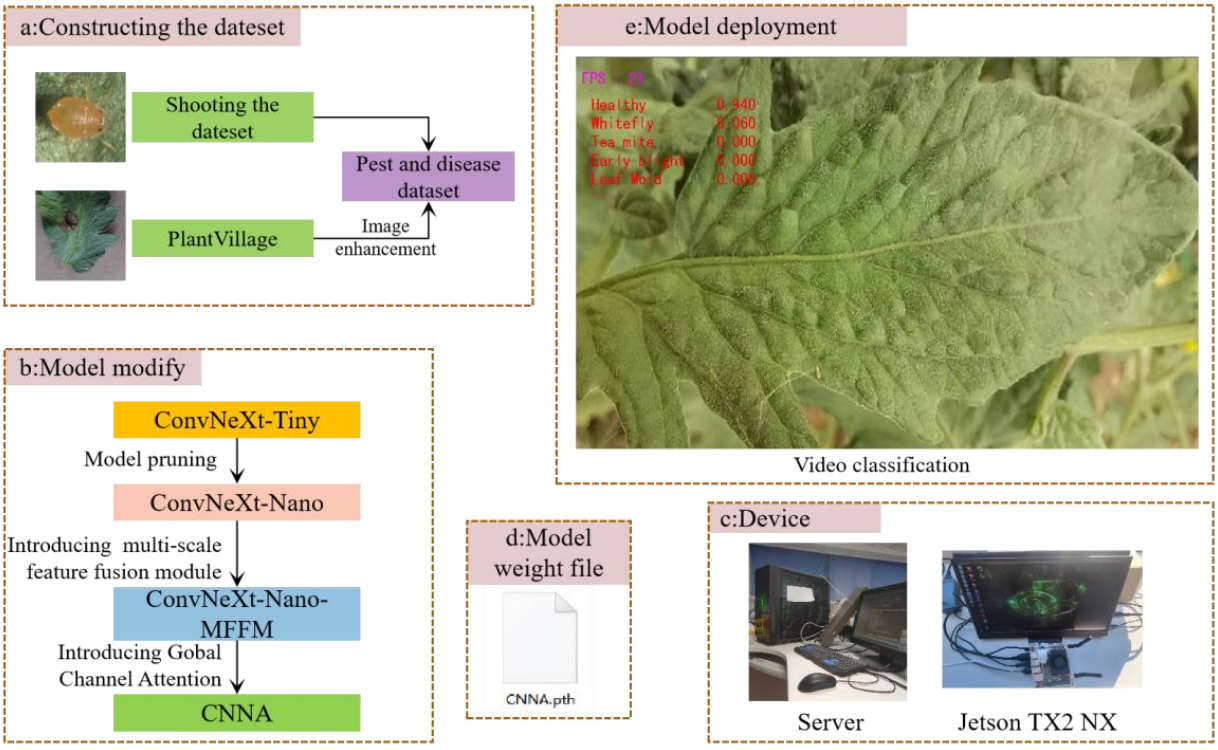


Figure 1. Workflow for the study: (**a**) constructing the image datasets, (**b**) model optimization, (**c**) device used in the study, (**d**) output model weight file, and (**e**) example of model deployment.

2.1. Dataset and Expansion

2.1.1. Acquisition of Images

In this study, we collected a total of 22,930 images from the PlantVillage dataset depicting both healthy and diseased plants, which were categorized into 10 classes, as shown in Figure 2a–j. In addition, we collected images of tomato leaves infested with pests to validate our proposed model from the experimental tomato field of Jilin Agricultural University in Changchun, Jilin Province, China. The images were collected between 9:00 a.m. and 5:00 p.m. using a smartphone (Xiaomi 10 in macro mode) with a resolution of 3120 × 3120 pixels. Finally, we obtained 431 images of tomato pests, belonging to 8 categories, with complex backgrounds, as shown in Figure 2k–r. Such images were preprocessed by eliminating images in which the pest subject was unclear, which were evaluated by agricultural experts. In addition, tomato pest and disease video stream data were collected according to the experimental requirements.

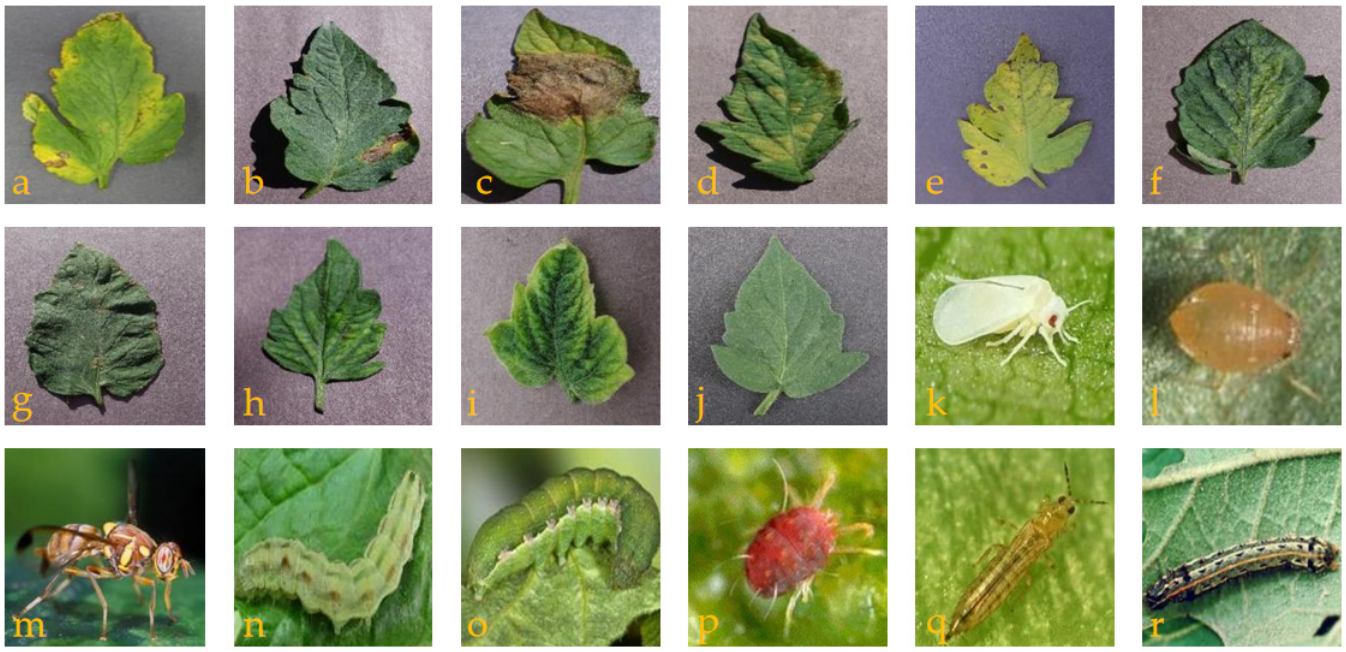


Figure 2. Tomato pest and disease dataset: (**a**) bacterial spot, (**b**) early blight, (**c**) late blight, (**d**) leaf mold, (**e**) septoria leaf spot, (**f**) two-spotted spider mite, (**g**) target spot, (**h**) mosaic virus, (**i**) tomato yellow leaf curl virus, (**j**) healthy leaf, (**k**) whitefly, (**l**) aphid, (**m**) American leaf miner, (**n**) cotton bollworm, (**o**) Diaphania indica, (**p**) spider leaf mite, (**q**) tea mite, and (**r**) tobacco budworm.

2.1.2. Image Preprocessing and Expansion

The tomato disease dataset was divided into training and validation sets in a ratio of approximately 80%: 20%. The resolution of all images was adjusted to 224 × 224 before data division to improve the efficiency of image processing. The distribution of the number of tomato disease images from PlantVillage for the training and validation sets is shown in Table 1.

Table 1. Distribution of tomato disease images in the open dataset.

Disease	Images	Training	Validation
Health	2407	1925	482
Bacterial spot	2127	1701	426
Early blight	2400	1920	480
Late blight	2314	1852	462
Leaf mold	2352	1882	470
Septoria leaf spot	2181	1745	432
Two-spotted spider mite	2176	1742	434
Target spot	2284	1828	456
Tomato mosaic virus	2238	1790	448
Tomato yellow leaf curl virus	2451	1961	490

The acquisition of tomato leaf pest images was more difficult, resulting in a smaller number of pest images compared to a larger number of tomato disease images in the open dataset. In order to avoid overfitting the model and improve its robustness, we augmented the original pest sample dataset by increasing its size approximately five-fold through two random 40-degree rotations and horizontal flips [26], which finally expanded the dataset to 4263 images. After enhancement, we constructed a tomato disease and pest dataset consisting of 27,193 images with 18 categories. The distribution of tomato pest images before and after the expansion is shown in Table 2.

Table 2. Tomato pest data enhancement.

Insect Pests	Number of Original Images	Number of Images after Random Enhancement	Training	Validation
Whitefly	61	420	336	42
Cotton bollworm	109	756	606	75
Aphid	131	910	728	91
Diaphania indica	76	525	421	52
Tobacco budworm	97	672	738	67
Tea mite	25	168	134	17
Spider leaf mite	75	518	141	52
American leaf miner	43	294	234	30

2.2. ConvNeXt-Nano-Adjust (CNNA) Network

2.2.1. Lightweight ConvNeXt-Nano and Other Variants

ConvNeXt, a convolutional neural network model proposed in 2021 [27], is a convolutional neural network model design based on ResNet50 [28] and the Swin transformer structure [29]. ConvNeXt-Tiny is the smallest version of ConvNeXt, but still suffers from the problem of model overfitting due to the stacking of modules and the excessive number of channels. To avoid the problem of saturating the model with disease and pest features due to the deep network layers of the ConvNeXt-Tiny network, we designed four variants of the ConvNeXt-Nano model based on the ConvNeXt-Tiny network by compressing the ConvNeXt model in two dimensions: the number of channels and the number of module stacks. ConvNeXt-Nano-1, ConvNeXt-Nano-2, ConvNeXt-Nano-3, and ConvNeXt-Nano were designed, and they were validated using images collected in the field to fully verify the effects of channel number and module stacking number on the global features. Additionally, we selected the optimal deep neural network based on model complexity and model performance, i.e., ConvNeXt-Nano.

All variants of ConvNeXt-Nano were created by reducing the number of modules in ConvNeXt-Tiny from four to three, reducing the operation by one layer, and then adjusting the number of module stacks and the number of channels per module. The number of module stacks in ConvNeXt-Tiny was [3,3,9,3] and the number of channels of the corresponding module was [96,192,384,768]. The number of module channels of

ConvNeXt-Nano-1 was adjusted to [48,96,192], and the number of module stacks was [3,9,3]. To further improve the accuracy, we used the control variable method to sep arately adjust the number of module stacks and the number of channels per layer to obtain ConvNeXt-Nano-2 and ConvNeXt-Nano-3. The number of module stacks of ConvNeXt-Nano-2 was adjusted to [2,6,2], and the number of module channels was [48,96,192], which was the same as ConvNeXt-Nano-1. ConvNeXt-Nano-3 was tested to adjust the number of channels per module to [24,48,96], and the number of module stacks was the same as ConvNeXt-Nano-1. ConvNeXt-Nano was tested to adjust the number of module stacks to [3,7,2], and the number of module channels to [24,48,96]. These improvements greatly reduced the numbers of network parameters and floating point operations (FLOPs). The above compression yielded the efficient lightweight network ConvNeXt-Nano, and the internal parameter comparison with the ConvNeXt-Tiny network is shown in Table 3.

Table 3. Internal parameters of the ConvNeXt-Nano model before and after compression.

Operator1	Operator2	Co1	Co2	n1	n2
Conv2d	Conv2d	3 → 96	3 → 24	1	1
LayerNorm	LayerNorm	96 → 96	24 → 24	1	1
ConvNeXt Block	ConvNeXt Block	96 → 96	24 → 24	3	3
ConvNeXt Block	ConvNeXt Block	96 → 192	24 → 48	3	7
ConvNeXt Block	ConvNeXt Block	192 → 384	48 → 96	9	2
ConvNeXt Block	-	384 → 768	-	3	-
GAP	GAP	768 → 768	96 → 96	1	1
LayerNorm	LayerNorm	768 → 768	96 → 96	1	1
Linear	Linear	768 → 18	96 → 18	1	1

The overall recognition of disease spots and pests by the ConvNeXt-Nano network was difficult due to the small number of lightweight network ConvNeXt-Nano parameters and the large variety of target species with different disease and pest characteristics in this study. To resolve this, we proposed a lightweight multi-scale feature fusion module (MFFM) and a lightweight global channel attention (GCA) mechanism. The CNNA network model optimizing the ConvNeXt-Nano network with MFFM and GCA is shown in the Figure 3.

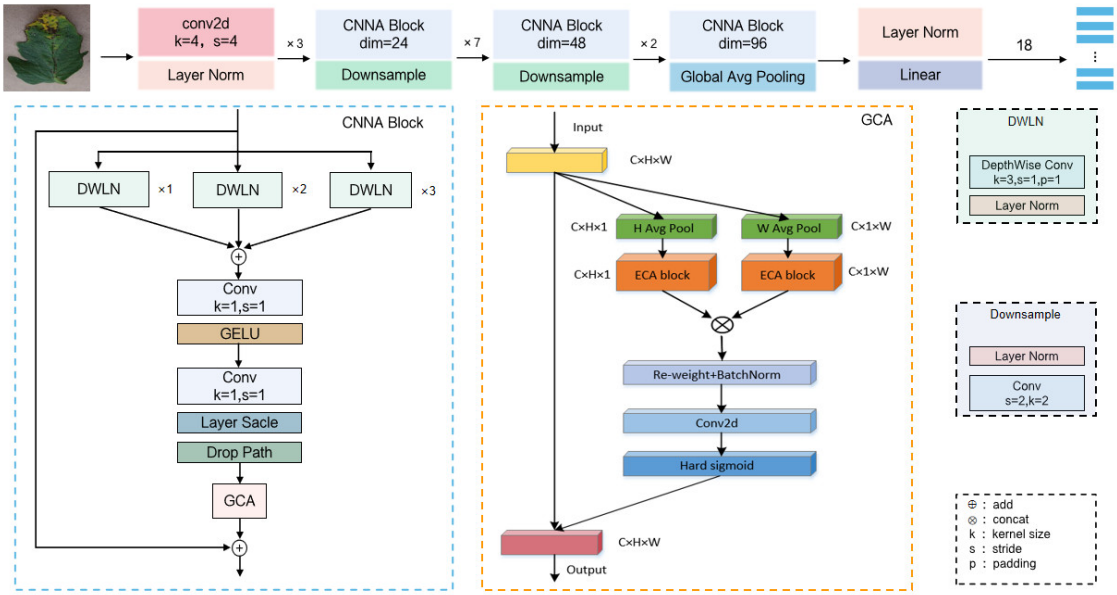


Figure 3. CNNA network structure (Layer Norm represents sample feature normalization) optimizing the ConvNeXt-Nano network with MFFM and GCA.

2.2.2. Multi-Scale Feature Fusion Module

Due to different pest sizes and the uneven distribution of leaf pests and diseases in the images, the feature distribution was uneven. As a result, it was hard to achieve high overall recognition. To improve the sensitivity of the CNNA model to features with different size, we proposed MFFM, as shown in Figure 4a. MFFM contained three branches, which were 3 × 3, 5 × 5, and 7 × 7 in depthwise convolution, as shown in Figure 4b. Depthwise separable convolution is a convolution operation with a small number of parameters and large memory access cost (MAC) [30], and multiple deep convolutions serially leads to increased inference time. Thus, using one large convolution kernel in deep convolution instead of multiple small convolution kernels in deep convolution can reduce the inference time. The model proposed in this study was named CNNA0 (CNNA0 compared to CNNA, only the structure of the multi-scale feature fusion module was replaced). Meanwhile, using multiple small convolution kernels in serial convolutions, compared with the large convolution kernel in convolutions [31], can reduce the number of model parameters and also reduce the picture inference time. The specific comparison experiments of the two schemes are described in Section 3.6. This study finally adopted the scheme of multiple small convolution kernels in depthwise convolutions instead of a large convolution kernel in depthwise convolutions. As shown in the dashed part of Figure 4, where DWLN represents one 3 × 3 depthwise convolution and feature normalization operation, two serial 3 × 3 depthwise convolutions instead of one 5 × 5 depthwise convolution for feature normalization, and three 3 × 3 depthwise convolutions instead of one 7 × 7 depthwise convolution.

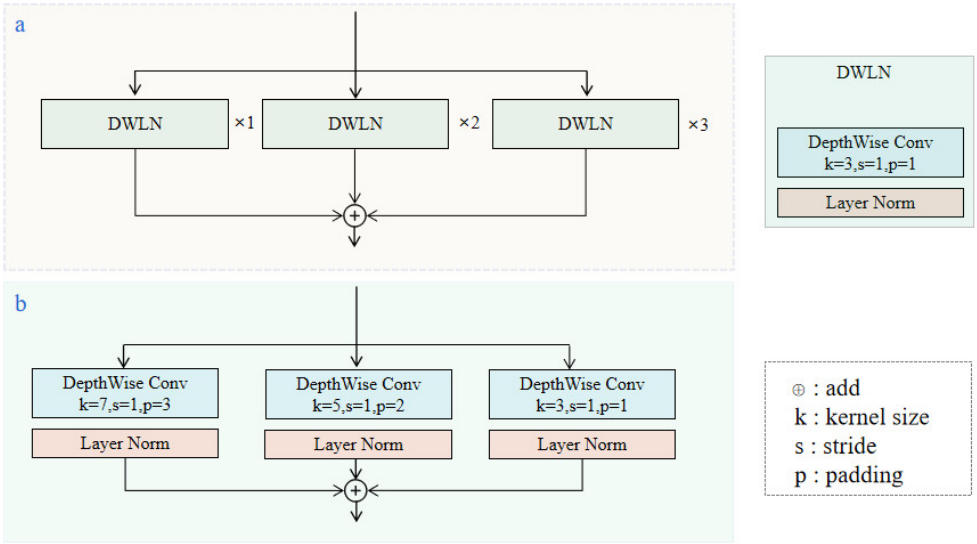


Figure 4. Multi-scale feature fusion module. ((**a**) represents the MFFM module stacked by multiple small convolution kernels, (**b**) represents the MFFM module stacked with a large convolution kernel).

2.2.3. Global Channel Attention for Optimizing the Model

The pest images with complex backgrounds in this experimental dataset were collected from the real-time field environment using a smartphone. In order to improve the global dependency of the CNNA model and increase its accuracy, this study proposed a lightweight global channel attention (GCA) mechanism, which encoded the channel relationship and long-term dependency by precise location information, fully integrating the information from the horizontal and vertical coordinates into the channel. Such a network formed a feature that was subject sensitive to direction and location information, which was beneficial to obtaining the pest subject information and suppressed useless information such as the background. The GCA structure is shown in Figure 5.

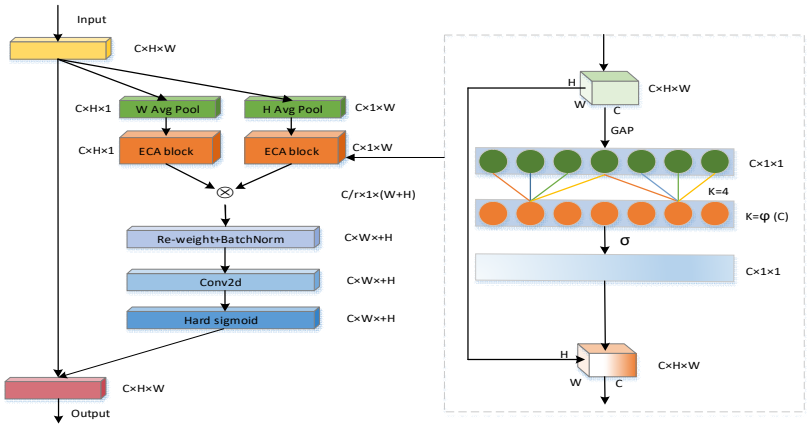


Figure 5. Global channel attention mechanism.

GCA encodes the adaptive global pooling method along the vertical and horizontal coordinates for each channel separately, and the input feature vector X is pooled using a convolutional kernel of size (H, 1) in the horizontal direction and a convolutional kernel of size (1, W) in the vertical direction. Equation (1) gives the vertical feature output value of the feature vector X of height c at the c-th channel:

$$Z_c^h(h) = \frac{1}{W} \sum_{0 \le i \le W} x_c(h, i) \tag{1}$$

where $Z_c^h(h)$ denotes the output of the c-th channel in the specific height direction, x_c denotes the feature vector of the input of the c-th channel, and W denotes the feature map width.

Equation (2) gives the horizontal feature output value of the feature vector X of width c at the c-th channel:

$$Z_c^w(w) = \frac{1}{H} \sum_{0 \le j \le H} x_c(j, w) \tag{2}$$

where $Z_c^w(w)$ denotes the output of the c-th channel in the specific width direction, and H denotes the feature map height. Equations (1) and (2) obtain the coded information at positions $C \times H \times 1$ and $C \times 1 \times W$, respectively, and then form a vector of $C \times 1 \times 1$ in two coordinate directions after the global averaging pooling operation without dimensionality reduction. Then, according to the idea of weight sharing of convolutional neural networks— i.e., the coverage of cross-channel information interaction (the size of the convolutional kernel of one-dimensional convolution) is proportional to the channel dimension C, which is proportional to the number of channels—the size of the adaptive convolutional kernel k is obtained, which calculates the weight of each channel and also reduces the number of parameters. Equation (3) gives the convolution kernel k of the computational formula:

$$k = \left| \frac{log_2(C)}{\gamma} + \frac{b}{\gamma} \right|_{odd} \tag{3}$$

where C denotes the number of channels; $|\ |_{odd}$ denotes that the convolution kernel k takes the nearest odd value; and γ and b are used to change the ratio between the number of channels C The two-dimensional feature maps are one-dimensionally stitched in the width direction, then reconstructed to the initial feature dimension and subjected to the feature normalization operation. By convolution, in order to extract the information of global feature fusion, this study used the lighter hard sigmoid activation function, as shown in Equation (4), where ReLU6 is the activation function and x is the dimension information of the input. The hard sigmoid activation function significantly reduced the computation and computation time compared to the sigmoid activation function because of the absence of

exponential operations. Finally, the global dimensional information was multiplied with the input dimensional information matrix for feature fusion.

$$\text{hard sigmoid}(x) = \frac{\text{ReLU6}(x+3)}{6} \tag{4}$$

2.3. Test Setup

To ensure fairness, the following experiments were conducted in Python 3.8 using the Pytorch framework, with the server using Pytorch-GPU 1.9 on Windows 10 and the Jetson TX2 NX using Pytorch 1.8 on Ubuntu 18.04 [32]. The server was powered by an Intel Core i7-7820X processor with 32 GB of RAM and NVIDIA TITAN Xp graphics with 12 GB of video memory. The Jetson TX2 NX had a CPU cluster consisting of a dual-core Denver2 processor and a quad-core ARM Cortex-A57, 4 GB of LDDR4 memory, and a 256-core Pascal GPU with power mode set to MAXN.

Each image was normalized before the input image, and the normalization formula shown in Equation (5) denotes the output after normalization and the input image derived from the results of large data training in ImageNet. The images needed to be computed in each of the three channels. Equation (5) denotes their mean values, which were taken as (0.485, 0.456, 0.406) for each of the three channels. The optimizer used AdamW (adaptive momentum and weight decay) with a cross-entropy loss function, the batch size was set to 64, the learning rate was initialized to 0.001, the number of rounds (epoch) was 50, and the learning rate declined by a factor of 0.1 every 10 rounds.

$$\text{output} = \frac{\text{input} - \text{mean}(\text{input})}{\text{std}(\text{input})} \tag{5}$$

2.4. Model Evaluation

To improve the performance evaluation of each model, we utilized precision, recall, F1 score, and accuracy as the evaluation indexes of the model to comprehensively evaluate the performance of the model, as shown in Equations (6)–(9).

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \times 100\% \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \times 100\% \tag{7}$$

$$\text{F1_score} = \frac{2\text{TP}}{2\text{TP}+\text{FP}+\text{FN}} \times 100\% \tag{8}$$

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \times 100\% \tag{9}$$

where TP, TN, FP, and FN are the number of true positive samples, true negative samples, false positive samples, and false negative samples, respectively. Precision is an estimate of how many of the predicted positive samples are positive, as shown in Equation (6). Recall is an assessment of how many of all positive samples can be correctly predicted as positive, as shown in Equation (7). F1 score is the summed average of the precision and recalls, as shown in Equation (8). Accuracy is the most intuitive measure of model quality, as shown in Equation (9). The model size, parameters, and floating point operations (*FLOPs*) are usually used to measure the complexity of the model. The *FLOPs* were calculated as shown in Equation (10).

$$FLOPs = \left[\sum_{L_{conv}=1}^{n}\left(2C_iK^2-1\right)HWC_O\right] + \left[\sum_{L_{full}=1}^{n}(2I-1)O\right] \tag{10}$$

where C_i is the number of input channels of the i-th convolutional layer; K is the convolutional kernel size; H and W are the height and width of the output feature map of the convolutional layer, respectively; C_O is the number of output channels of the convolutional layer; and I and O are the number of inputs and outputs in the fully connected layer, respectively.

3. Validation and Results

3.1. Complexity and Performance Comparison of CNNA Model Variants

Too many layers of network stacking can cause model overfitting and lead to accuracy degradation. This experiment was conducted to compare the number of parameters and model performance before and after model pruning. The experiments and results are shown in Table 4.

As can be seen in Table 4, ConvNeXt-Nano-1 reduced the number of parameters from 27.83 to 1.79 M and the number of FLOPs from 4457.49 to 964.02 M compared to ConvNeXt-Tiny, which resulted in only a loss of 0.37% accuracy with a significant reduction in the numbers of parameters as well as FLOPs. In addition, the results proved that ConvNeXt-Tiny network had the problem of model overfitting due to excessive module stacking. ConvNeXt-Nano-2 reduced module stacking by 5 times compared to ConvNeXt-Nano-1, and the numbers of parameters and FLOPs were reduced by 31% and 32%, respectively. Additionally, the accuracy was improved by 0.22%. Similarly, ConvNeXt-Nano-3 improved the accuracy compared to ConvNeXt-Nano-1, and the number of channels per layer was reduced by a factor of 1. At the same time, the numbers of parameters and FLOPs were reduced by 74% and 73%, respectively, and the accuracy was improved by 0.54%. The experimental results proved that the reduction in the number of module stacks did not have a significant impact on the accuracy, and the increase in the number of channels per layer led to an increase in the numbers of model parameters and FLOPs, which also had a greater impact on the accuracy. In general, ConvNeXt-Nano was a variant that kept the number of channels the same as ConvNeXt-Nano-3 while adjusting the module stacks to [3,7,2] and the numbers of parameters and FLOPs. CNNA further improved ConvNeXt-Nano by introducing MFFM and GCA into the structure of ConvNeXt-Nano, and the numbers of model parameters and FLOPs increased by 6% and 13%, respectively, while the accuracy was greatly improved by 2.65%. Thus, this variant had the best comprehensive performance.

Table 4. Performance comparison of network parameters before and after compression.

Network	Dims	Depths	Parameters/M	FLOPs/M	Accuracy/%
ConvNeXt-Tiny	(3,3,9,3)	(96,192,384,768)	27.83	4457.49	96.28
ConvNeXt-Nano-1	(3,9,3)	(48,96,192)	1.79	964.02	95.91
ConvNeXt-Nano-2	(2,6,2)	(48,96,192)	1.23	654.73	96.13
ConvNeXt-Nano-3	(3,9,3)	(24,48,96)	0.47	258.03	96.35
ConvNeXt-Nano	(3,7,2)	(24,48,96)	0.35	210.06	96.31
CNNA	(3,7,2)	(24,48,96)	0.37	237.61	98.96

3.2. Performance Evaluation of Different Attention Mechanisms Models

To verify the performance of the SE, CA [33], ECA [34], and GCA proposed in this paper, the attention mechanisms were introduced into the CNNA network in the same way for performance comparison experiments, and the results are shown in Table 5.

Table 5. Performance comparison of introducing different attention mechanisms.

Attention Mechanisms	Accuracy/%	Precision/%	Recall/%	F1-Score/%	Parameters/M	FLOPs/M
None	96.31	94.33	93.22	93.73	0.36	210.06
SE	97.68	96.28	96.86	97.05	0.37	227.35
CA	97.77	96.01	95.46	95.70	0.38	229.60
ECA	98.31	97.51	97.11	97.29	0.37	228.50
GCA	98.96	97.65	97.47	97.53	0.37	237.61

As shown in Table 6, the accuracy after introducing an attention mechanism was significantly improved compared to that without introducing an attention mechanism, which verified the necessity of introducing the attention mechanism. The model complexity after introducing SE and CA was almost comparable, and the accuracy of introducing CA was 0.09% higher than that of introducing SE, but the accuracy, recall, and F1 score of CA were lower than those of SE. The index results of introducing ECA were better than those of introducing SE and CA, and the accuracy of introducing ECA was 0.63% and 0.54% higher than that of introducing CA and SE, respectively. This may have been related to the efficient channel dependence of ECA and weight-sharing learning. Meanwhile, introducing GCA into the model achieved the highest accuracy, precision, recall, and F1 score, while the number of parameters was minimal. The accuracy was improved by 0.65% compared to introducing ECA, which verified that GCA was more capable of acquiring global features.

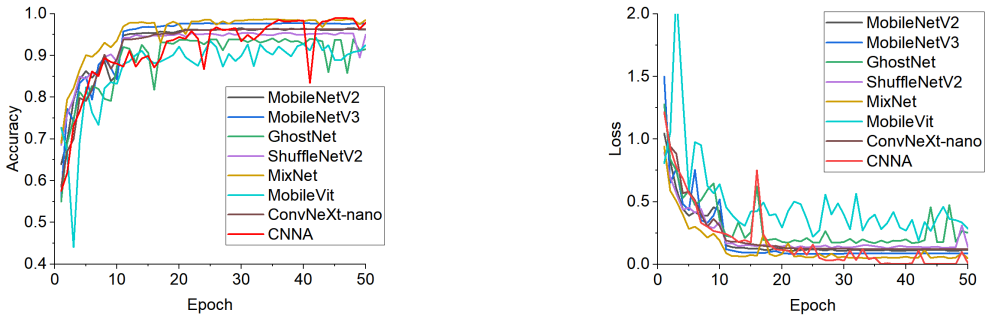
Table 6. Performance comparison with other lightweight convolutional neural networks.

Network	Accuracy /%	Precision /%	Recall /%	F1-Score /%	Parameters /M	FLOPs /M	Size /MB
MobileNetV2	96.63	92.76	92.13	92.38	2.25	318.98	8.80
MobileNetV3	97.77	95.36	94.84	95.07	4.23	226.45	16.26
GhostNet	94.14	85.27	84.11	84.44	3.14	216.62	12.21
ShuffleNetV2	95.51	91.90	89.84	90.63	1.27	149.59	4.96
MixNet	98.32	97.43	97.11	97.29	2.62	250.31	10.28
MobileVit	94.30	93.74	92.37	93.11	4.95	1464.38	19.90
ConvNeXt-Nano	96.31	94.33	93.22	93.73	0.35	190.99	1.37
CNNA	98.96	97.85	97.57	97.76	0.37	237.61	1.47

3.3. Performance Comparison with Other Lightweight Network Parameters

To validate the parameters and performance of the CNNA network, based on the dataset constructed in this study, the ConvNeXt-Nano and CNNA networks were compared with some of the more widely used and effective lightweight networks in this study. The compared networks were MobileNetV2 [35], MobileNetV3 [36], GhostNet [37], ShuffleNetV2 [30], MixNet [38], and MobileVit [39]. As shown in Table 6, the CNNA network significantly outperformed the other networks in all evaluation metrics. The recognition accuracy of GhostNet and MobileVit was relatively low, at 94.14% and 94.30%, respectively. Compared with the GhostNet and MobileVit models, the recognition accuracy of ConvNeXt-Nano was higher, reaching 96.31% and proving the effectiveness of network compression and optimization. The recognition accuracy of the CNNA network reached 98.96%, with a small increase in the numbers of parameters and FLOPs compared with ConvNeXt-Nano but a 2.01% accuracy improvement, proving the effectiveness of introducing MFFM and GCA. Compared with MixNet, the CNNA model had 85% fewer parameters, 5% fewer FLOPs, 86% smaller model size, and 0.64% higher accuracy. In summary, among the seven lightweight models compared, except ConvNeXt-Nano, the CNNA network model had the smallest number of parameters, the smallest model, and fewer FLOPs, but it achieved the best convergence and the highest accuracy.

To further verify the performance and parameters of the CNNA network, the accuracy and loss curves of six lightweight networks, ConvNeXt-Nano, and the CNNA network were plotted in this study using the validation set of tomato pest data, as shown in Figure 6. The accuracy of each model on the validation set tended to be stable after 50 rounds of iterations. The accuracy curves showed that the CNNA network had the highest recognition accuracy but there was a sudden decrease and then an increase in accuracy. This change may have been caused by the large difference between the pest subject and the background environment of the different pests in the images, since the CNNA network learned less information about the same kinds of pest features. This led to the phenomenon that the model had errors in recognizing the same kind of pest, but this did not affect its overall recognition accuracy of tomato pests and diseases. The loss curve showed that the loss value of the CNNA network kept decreasing in the first 10 rounds and then started to converge in 10 rounds, but there was an increase in the middle. Additionally, the last 30 rounds tended to smooth out and almost converged to 0. In summary, the improved CNNA network achieved higher accuracy and lower loss than the other networks.



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

Figure 6. Accuracy and loss curves. (**a**) Accuracy curve of the validation set; (**b**) loss curve of the validation set.

3.4. Ablation Experiments

To verify the performance improvement achieved by introducing model compression optimization, MFFM, and GCA into the CNNA model, ablation experiments were conducted to introduce MFFM and GCA into the ConvNeXt-Nano network. The results are shown in Table 7.

Table 7. Experimental results of ablation of different modules.

Model	Factors			Accurary/%	F1-Score/%	Parameters/M
	Model Compression	Multi-Scale Feature Fusion Module (MFFM)	Global Channel Attention (GCA)			
ConvNeXt-Tiny	×	×	×	96.28	95.26	27.81
ConvNeXt-Nano	√	×	×	96.31	93.73	0.35
	√	√	×	97.17	94.65	0.36
	√	×	√	97.76	96.15	0.36
CNNA	√	√	√	98.96	97.76	0.37

Analysis of Table 7 shows that the addition of the MFFM to the ConvNeXt-Nano network increases the number of parameters by only 0.01 M, but the accuracy and F1 score increase by 0.86% and 0.92%, respectively. The introduction of this module demonstrated that the network was able to extract information on different size pest features to some extent. Embedding only GCA was able to efficiently incorporate the global coordinate information into the channels, and both accuracy and F1 score were improved. Finally, by

introducing the improved strategy of multi-scale feature fusion module and global channel attention, the accuracy and F1 score of CNNA on the tomato diseases and pests validation set were 98.96% and 97.76%, which were 2.65% and 4.23% better than ConvNeXt-Nano, and 75 times lower than ConvNeXt-Tiny model parametric number, while the accuracy and F1 score have 2.68% and 2.50% improvement. Analyzing the confusion matrix plot, as shown in Figure 7, most of the pests and diseases were correctly predicted by the CNNA model in the case of unbalanced dataset 15 images of cotton bollworm were incorrectly predicted, 7 images of late blight were predicted as early blight, and no other prediction error categories exceeded 10, which proved that the CNNA model has better recognition and generalization ability.

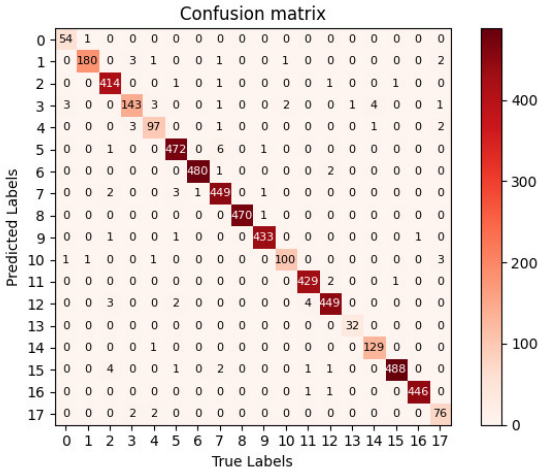


Figure 7. Confusion matrix (where label 0 corresponds to American leaf miner, 1 corresponds to aphid, 2 corresponds to bacterial spot, 3 corresponds to cotton bollworm, 4 corresponds to Diaphania indica, 5 corresponds to early blight, 6 corresponds to healthy leaves, 7 corresponds to late blight, 8 corresponds to leaf mold, 9 corresponds to septoria leaf spot, 10 corresponds to spider leaf mite, 11 corresponds to two-spotted spider mite, 12 corresponds to target spot, 13 corresponds to tea mite, 14 corresponds to tobacco budworm, 15 corresponds to tomato yellow leaf curl virus, 16 corresponds to tomato mosaic virus, and 17 corresponds to whitefly).

3.5. Network Attention Visualization

To better observe the learning ability of the CNNA model proposed in this experiment for tomato pest features, the classification results were visualized using Grad-CAM [40]. Specifically, feature visualization was performed on some data samples of the validation set, as shown in Figure 8. The MixNet, ConvNeXt-Nano, and CNNA network models were used for comparison, and the last layer was used for the network feature visualization layer. From the figure, it can be seen that the heatmap of MixNet focused on small disease patches and local areas of insect infestation, and the accuracy of the heatmap was not surprisingly high. The heatmap of ConvNeXt-Nano focused on patch areas but contained much irrelevant background information and could not identify whitefly. Compared with the MixNet and ConvNeXt-Nano models, the heatmap of CNNA could accurately identify key regions of late blight, with high accuracy and less attention paid to irrelevant and complex backgrounds. Meanwhile, the addition of MFFM with small-sized convolutional kernels resulted in more accurate whitefly identification in small regions.

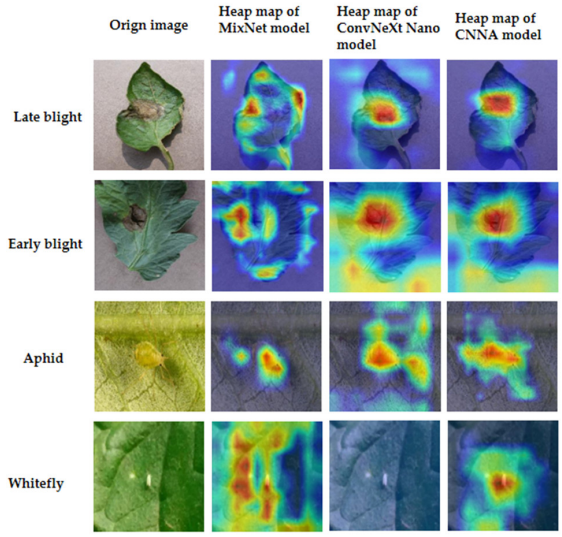


Figure 8. Comparison table of Grad-CAM heatmaps.

3.6. Model Deployment and Comparison

To further validate the performance of the CNNA model for identifying tomato pests and diseases and the inference time of the CNNA0 network mentioned in Section 2.2.2, the MixNet model with high accuracy, the MobileNetV3 model with high reputation, the CNNA0 model with multi-scale modules using large convolutional kernels, and the CNNA model proposed in this study were deployed to the Jetson TX2 NX and the server, respectively. Image inference was performed on the constructed tomato pest and disease validation set, and the inference time for a single image was calculated. Then, this study imported the captured video stream data into the Jetson TX2 NX, which performed classification based on the video data and displayed the maximum possible pest categories on the monitor (Figure 9). The comparison of single image inference time, numbers of model parameters and FLOPs, and accuracy are shown in Table 8, where the test results of a single image inference time are reported as the average of five experiments.

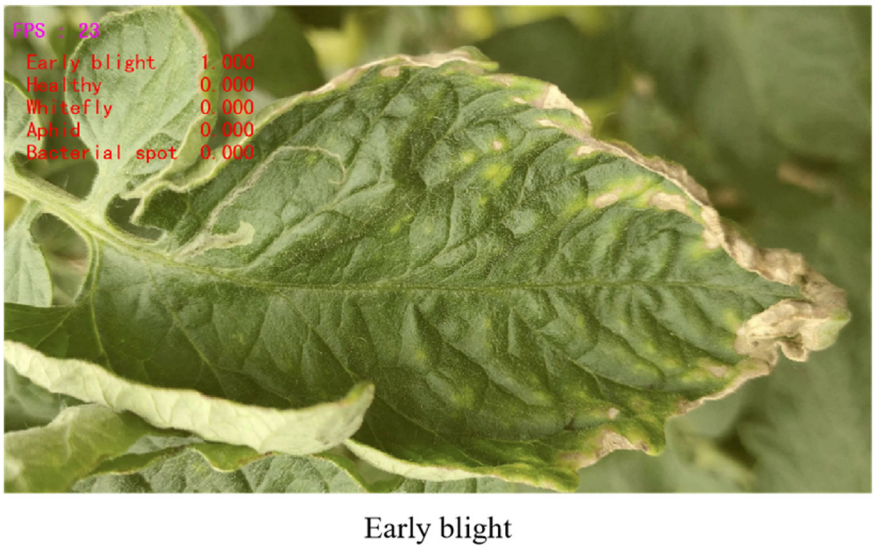


Figure 9. Jetson TX2 NX video inference. (FPS represents the frame rate of video inference, while the red letters represent the five categories inferred by the model from the image with the highest probability and the probability of the model predicting that category).

Table 8. Performance comparison of different models in TITAN Xp and Pascal.

Model	Pascal/ms	TITAN Xp/ms	Parameters/M	FLOPs/M	Accuracy/%
MixNet	63.22	13.91	4.23	226.45	98.32
MobileNetV3	38.24	11.51	2.62	250.31	97.77
CNNA0	48.61	12.66	0.41	252.88	98.96
CNNA	47.35	11.42	0.37	237.61	98.96

From Table 8, it can be seen that MixNet had the largest number of parameters, performed the worst in both Pascal and TITAN Xp, and took the longest time to reason one image. The average inference times were 47.35 ms in Pascal and 11.42 ms in TITAN Xp. These results proved that the depth-separable convolutional model was faster than CNNA0 in the case of a relatively small model and a small number of serial depth-separable convolutions. In addition, the inference speed was weighted more on the numbers of model parameters and FLOPs.

4. Discussion and Conclusions

In this study, a lightweight pest and disease classification model was developed for 18 categories of tomato leaves. The experimental data used open-source images of 10 classes of diseased and healthy tomato leaves in PlantVillage (Figure 2a–j). Images of eight insect pests on tomato leaves were taken in an experimental field to validate the proposed deep neural network (Figure 2k–r). In addition, the dataset of tomato pest image data was enhanced to enrich the data sample features.

We compressed and optimized the structure of the ConvNeXt-Tiny network by pruning the original four modules to three while reducing the number of channels of each module, thus optimizing it into a lightweight convolutional neural network. We added MFFM to improve the model's recognition ability for different disease spot sizes and pests. Additionally, we embedded GCA to incorporate global feature information into the channels and enhance the network model. The CNNA network had a parameter count of 0.37 M, 237.61 M FLOPs, and a model size of 1.47 MB, which were 75, 18, and 72 times smaller than those of ConvNeXt-Tiny, respectively. Meanwhile, the error rate of the model was only 1% and the accuracy was 98.96%, further demonstrating its outstanding performance.

Furthermore, the CNNA model was deployed into the edge intelligence device, the Jetson TX2 NX, which performed inference recognition based on the video stream data and displayed the maximum possible pest categories on the monitor, with an average inference time of 47.35 ms for one image. This verified the superiority of the model in inference time and provided technical support for the development of tomato pest and disease control system.

In summary, the CNNA model can perform the classification of tomato pest and disease image data quickly and efficiently, thereby assisting agriculture-related personnel in improving production efficiency, reducing labor intensity, and decreasing the use of pesticides and other chemicals. This model is also applicable to non-tomato data and can be extended to tasks such as target detection and image segmentation. In future work, we will install the Jetson TX2 NX device in the experimental tomato field of Jilin Agricultural University and build a real-time tomato pest and disease classification platform. We will collect tomato leaf video data using a mobile USB camera, and the Jetson TX2 NX will perform real-time classification and identification based on the video to display the maximum possible pest and disease categories for relevant processing by the lower computer. Furthermore, we will acquire more tomato pest images with complex backgrounds in the field as input for model training in order to improve the model's ability to generalize tomato pests and diseases in complex backgrounds.

Author Contributions: Conceptualization, Y.X. and Z.G. (Zhiyuan Gao); Methodology, Y.X. and Y.Z. (Yuting Zhai); Validation, Z.G. (Zhiyuan Gao) and Y.Z. (Yang Zhou); Writing—Original Draft Preparation, Z.G. (Zhiyuan Gao); Writing—Review & Editing, Q.W., Z.G. (Zongmei Gao) and Y.Z.

(Yuting Zhai); Supervision, Z.X. and Y.X.; Funding Acquisition, Y.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Science and Technology Development Plan Project of Changchun [grant number 21ZGN28]; the Jilin Provincial Science and Technology Development Plan Project [grant number 20230202035NC]; and the Jilin Province Science and Technology Development Plan Project [grant number YDZJ202301ZYTS408].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are not publicly available due to privacy restrictions.

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

References

1. Picó, B.; Díez, M.J.; Nuez, F. Viral diseases causing the greatest economic losses to the tomato crop. II. The Tomato yellow leaf curl virus—A review. *Sci. Hortic.* **1996**, *67*, 151–196. [CrossRef]
2. Moretti, C.; Bocchini, M.; Quaglia, M.; Businelli, D.; Orfei, B.; Buonaurio, R. Sodium Selenate: An Environmental-Friendly Means to Control Tomato Bacterial Speck Disease. *Agronomy* **2022**, *12*, 13. [CrossRef]
3. Hong, S.-J.; Park, J.-H.; Kim, Y.-K.; Jee, H.-J.; Han, E.-J.; Shim, C.-K.; Kim, M.-J.; Kim, J.-H.; Kim, S.-H. Study on the control of leaf mold, powdery mildew and gray mold for organic tomato cultivation. *Korean J. Org. Agric.* **2012**, *20*, 655–668. [CrossRef]
4. Ebrahimi, M.A.; Khoshtaghaz, M.H.; Minaei, S.; Jamshidi, B. Vision-based pest detection based on SVM classification method. *Comput. Electron. Agric.* **2017**, *137*, 52–58. [CrossRef]
5. Bharate, A.A.; Shirdhonkar, M. A review on plant disease detection using image processing. In Proceedings of the 2017 International Conference on Intelligent Sustainable Systems (ICISS), Palladam, India, 7–8 December 2017; pp. 103–109.
6. Deng, W.Z.; Zhou, F.Y.; Gong, Z.; Cui, Y.J.; Liu, L.; Chi, Q. Disease Feature Recognition of Hydroponic Lettuce Images Based on Support Vector Machine. *Trait. Signal* **2022**, *39*, 617–625. [CrossRef]
7. Mokhtar, U.; Ali, M.A.; Hassanien, A.E.; Hefny, H. Identifying two of tomatoes leaf viruses using support vector machine. In *Information Systems Design and Intelligent Applications: Proceedings of Second International Conference INDIA 2015*; Springer: Berlin/Heidelberg, Germany, 2015; Volume 1, pp. 771–782.
8. Johannes, A.; Picon, A.; AlVarez-Gila, A.; Echazarra, J.; Rodriguez-Vaamonde, S.; Navajas, A.D.; Ortiz-Barredo, A. Automatic plant disease diagnosis using mobile capture devices, applied on a wheat use case. *Comput. Electron. Agric.* **2017**, *138*, 200–209. [CrossRef]
9. Rumpf, T.; Mahlein, A.-K.; Steiner, U.; Oerke, E.-C.; Dehne, H.-W.; Plümer, L. Early detection and classification of plant diseases with support vector machines based on hyperspectral reflectance. *Comput. Electron. Agric.* **2010**, *74*, 91–99. [CrossRef]
10. Aggarwal, S.; Gupta, S.; Gupta, D.; Gulzar, Y.; Juneja, S.; Alwan, A.A.; Nauman, A. An Artificial Intelligence-Based Stacked Ensemble Approach for Prediction of Protein Subcellular Localization in Confocal Microscopy Images. *Sustainability* **2023**, *15*, 20. [CrossRef]
11. Yang, G.F.; Chen, G.P.; He, Y.; Yan, Z.Y.; Guo, Y.; Ding, J. Self-Supervised Collaborative Multi-Network for Fine-Grained Visual Categorization of Tomato Diseases. *IEEE Access* **2020**, *8*, 211912–211923. [CrossRef]
12. Ji, M.; Zhang, L.; Wu, Q. Automatic grape leaf diseases identification via UnitedModel based on multiple convolutional neural networks. *Inf. Process. Agric.* **2020**, *7*, 418–426. [CrossRef]
13. Anandhakrishnan, T.; Jaisakthi, S.M. Deep Convolutional Neural Networks for image based tomato leaf disease detection. Sustain. *Chem. Pharm.* **2022**, *30*, 11. [CrossRef]
14. Gao, X.; Ramezanghorbani, F.; Isayev, O.; Smith, J.S.; Roitberg, A.E. TorchANI: A Free and Open Source PyTorch-Based Deep Learning Implementation of the ANI Neural Network Potentials. *J. Chem. Inf. Model.* **2020**, *60*, 3408–3415. [CrossRef] [PubMed]
15. Borhani, Y.; Khoramdel, J.; Najafi, E. A deep learning based approach for automated plant disease classification using vision transformer. *Sci. Rep.* **2022**, *12*, 10. [CrossRef] [PubMed]
16. Hassan, S.K.M.; Jasinski, M.; Leonowicz, Z.; Jasinska, E.; Maji, A.K. Plant Disease Identification Using Shallow Convolutional Neural Network. *Agronomy* **2021**, *11*, 20. [CrossRef]
17. Gulzar, Y. Fruit Image Classification Model Based on MobileNetV2 with Deep Transfer Learning Technique. *Sustainability* **2023**, *15*, 14. [CrossRef]
18. Elhassouny, A.; Smarandache, F. Smart mobile application to recognize tomato leaf diseases using Convolutional Neural Networks. In Proceedings of the 2019 International Conference of Computer Science and Renewable Energies (ICCSRE), Agadir, Morocco, 22–24 July 2019; pp. 1–4.
19. Agarwal, M.; Singh, A.; Arjaria, S.; Sinha, A.; Gupta, S. ToLeD: Tomato leaf disease detection using convolution neural network. *Procedia Comput. Sci.* **2020**, *167*, 293–301. [CrossRef]
20. Wang, G.; Sun, Y.; Wang, J.X. Automatic Image-Based Plant Disease Severity Estimation Using Deep Learning. *Comput. Intell. Neurosci.* **2017**, *2017*, 8. [CrossRef]

21. Hamid, Y.; Wani, S.; Soomro, A.B.; Alwan, A.A.; Gulzar, Y. Smart seed classification system based on MobileNetV2 architecture. In Proceedings of the 2022 2nd International Conference on Computing and Information Technology (ICCIT), Karachi, Pakistan, 15–16 January 2022; pp. 217–222.
22. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E.H. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef]
23. 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.
24. Yin, C.H.; Zeng, T.W.; Zhang, H.M.; Fu, W.; Wang, L.; Yao, S.Y. Maize Small Leaf Spot Classification Based on Improved Deep Convolutional Neural Networks with a Multi-Scale Attention Mechanism. *Agronomy* **2022**, *12*, 18. [CrossRef]
25. Gao, R.H.; Wang, R.; Feng, L.; Li, Q.F.; Wu, H.R. Dual-branch, efficient, channel attention-based crop disease identification. *Comput. Electron. Agric.* **2021**, *190*, 10. [CrossRef]
26. Mamat, N.; Othman, M.F.; Abdulghafor, R.; Alwan, A.A.; Gulzar, Y. Enhancing Image Annotation Technique of Fruit Classification Using a Deep Learning Approach. *Sustainability* **2023**, *15*, 19. [CrossRef]
27. Liu, Z.; Mao, H.; Wu, C.-Y.; Feichtenhofer, C.; Darrell, T.; Xie, S. A convnet for the 2020s. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 11976–11986.
28. He, K.M.; Zhang, X.Y.; Ren, S.Q.; 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.
29. Liu, Z.; Lin, Y.T.; Cao, Y.; Hu, H.; Wei, Y.X.; Zhang, Z.; Lin, S.; Guo, B.N. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11–17 October 2021; pp. 10012–10022.
30. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 116–131.
31. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
32. Zhang, W.L.; Liu, Y.X.; Chen, K.Z.; Li, H.B.; Duan, Y.L.; Wu, W.B.; Shi, Y.; Guo, W. Lightweight Fruit-Detection Algorithm for Edge Computing Applications. *Front. Plant Sci.* **2021**, *12*, 16. [CrossRef]
33. Hou, Q.B.; Zhou, D.Q.; Feng, J.S. Coordinate Attention for Efficient Mobile Network Design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 13713–13722.
34. Wang, Q.L.; Wu, B.G.; Zhu, P.F.; Li, P.H.; Zuo, W.M.; Hu, Q.H. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11534–11542.
35. Howard, A.; Zhmoginov, A.; Chen, L.-C.; Sandler, M.; Zhu, M.L. Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation. *arXiv* **2018**, arXiv:1801.04381.
36. Howard, A.; Sandler, M.; Chu, G.; Chen, L.-C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V. Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
37. Han, K.; Wang, Y.H.; Tian, Q.; Guo, J.Y.; Xu, C.J.; Xu, C. GhostNet: More Features from Cheap Operations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2020; pp. 1580–1589.
38. Tan, M.; Le, Q.V. MixConv: Mixed Depthwise Convolutional Kernels. *arXiv* **2019**, arXiv:1907.09595.
39. Mehta, S.; Rastegari, M. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv* **2021**, arXiv:2110.02178.
40. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.

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.