Precision Agriculture (2023) 24:1836–1861
https://doi.org/10.1007/s11119-023-10020-0

An ultra-lightweight efficient network for image-based plant disease and pest infection detection

Beibei Wang[1] · Chenxiao Zhang[2] · Yanyan Li[3] · Chunxia Cao[1] · Daye Huang[1] · Yan Gong[1]

Accepted: 29 March 2023 / Published online: 22 April 2023

Abstract
Plant diseases and pest infections are major factors that undermine the growth of plants along with their life cycle. Optical image-based plant disease detection provides an efficient and low cost way for real-time plant growth monitoring and management. In recent years, the thriving development of deep learning techniques in a variety of communities has validated its great performance in image interpretation and understanding. Existing deep learning-based methods for plant disease classification mostly adopt convolutional neural networks (CNNs) that have been originally developed for general image classification purposes. These CNN architectures consist of a very large volume of training parameters, which severely hinders its applicability under scenarios requiring fast and flexible deployment on compact devices with limited computation powers. In this paper, an ultra-lightweight efficient network (ULEN) is proposed targeting image-based plant disease and pest infection detection. The proposed network consists of two parts, a deep feature extraction module that adopts residual depth-wise convolution and a classification module receiving multi-scale features enhanced by a spatial pyramid pooling layer. The network is constructed in a very compact design with approximately only 100 000 parameters, which greatly favors the demand for a lightweight model for practical needs. Two publicly available plant datasets collected at the indoor and outdoor environments were tested on two compact devices to validate its applicability under different scenarios. Compared with the state-of-the-art architectures, the proposed network showed superior performance with the least computation complexity and compelling classification accuracy.

Keywords Plant disease detection · Pest infection detection · Convolutional neural network · Deep learning

✉ Yan Gong
gongyan@nberc.com

[1] National Biopesticide Engineering Research Centre, Hubei Biopesticide Engineering Research Centre, Hubei Academy of Agricultural Sciences, Wuhan 430064, China

[2] School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430072, China

[3] Tobacco Research Institute of Hubei Province, Wuhan 430030, China

Introduction

According to the estimates by the Food and Agriculture Organization (FAO) ("Scientific review of the impact," 2021), plant pests that ravage economically important crops are becoming more destructive due to the impact of climate change, resulting in a huge reduction of 40% global crop production every year. Moreover, plant diseases cost more than 220 billion U.S. dollars around the globe. The occurrence and extent of plant diseases and pests have significantly damaged the quality of agricultural production and crop yields and posed an increasing threat to food security (FAO, 2021). The timely and accurate detection of plant diseases and pest infection could provide sufficient and intuitive information to support management decisions (e.g., pesticide application), thus playing a key role in agriculture (Miller et al., 2009).

Early image recognition-based methods refer to low-level image features (e.g., scale-invariant feature transform) with machine learning classification algorithms (e.g., random forest) to realize plant image classification (Hlaing and Zaw 2017; Rançon et al., 2019). These methods have released the plant disease detection task from expertise knowledge involvement and have greatly improved the efficiency of plant disease and pest infection detection. However, they heavily rely on manually crafted features that are not capable of representing disease-dependent features that is helpful for image understanding under complex environments. In recent years, deep learning techniques have been widely utilized in a variety of communities including the agriculture community, inspiring some insightful deep learning-based applications (Kamilaris & Prenafeta-Boldú, 2018) and continuously expanding its potential for precision agriculture.

Among the growing developed architectures in deep learning, convolutional neural networks (CNNs) haved raised the most interest in agriculture. Contributing to the concept of shared convolutional kernels and stacked convolutional layers and pooling layers, CNN can hierarchically extract low and high-level features that are barely encoded by artificially designed algorithms, thus achieving a strong feature representation ability. Therefore, CNN-based networks are becoming the most commonly used architecture in plant disease and pest infection classification. Tested on a publicly accessible dataset Plantvillage (Hughes & Salathé, 2015), LeNet and GoogLeNet were used to validate the effectiveness of stacked CNNs for plant disease image classification (Mohanty et al., 2016). Another initial research also applied LeNet5 architecture with two convolutional layers and three fully connected layers for low-resolution maize leaf disease detection (Priyadharshini et al., 2019). It should be noted that LeNet5 was firstly proposed in 1989 (LeCun et al., 1989). A modified CaffeNet architecture with five convolutional and three fully connected layers was used for plant disease image classification on a manually collected dataset consisting of fifteen plant disease classes (Sladojevic et al., 2016). Since networks pre-trained on large-scale datasets have already gained a strong representation ability, studies on transferring the pre-trained networks for image-based plant disease detection have been conducted. An Xception-based architecture pretrained on the ImageNet (Deng et al., 2009) was transferred and fine tuned for tomato lead disease classification (Thangaraj et al., 2021). EfficientNet has been proposed for plant disease detection, achieving 99% accuracy on the Plantvillage dataset (Atila et al., 2021). Insisting that high-resolution images can help to boost classification performance, generative adversarial network was first used to rebuild high-resolution images from low-resolution images. Then the VGG16 model was applied for crop disease classification (Wen et al., 2020). Comparisons of performance and model compactness between AlexNet, VGGNet, Inception, ResNet and MobileNet have been

conducted on Plantvillage Dataset. The proposed models were modified based on the four architectures and achieved an average precision of 98.6% with a minimum number of training parameters of 3,370,000 (Hassan et al., 2021). To the best of the authors' knowledge, this is the most compact model proposed for the plant disease and pest infection task. Similarly, Singh et al. (2021) compared the classification performances of VGGNet, Inception, DenseNet, MobileNet, Xception and NASNetMobile for disease and pest infection detection in coconut trees. The experimental results showed that MobileNet achieved the best classification accuracy of 82.1%. In the literature, several advanced networks have been applied to classify plant disease images. It has been acknowledged that with the network going deeper, more representative features can be learned so that the network performance can be boosted (Szegedy et al., 2015). However, the gain in performance can be quite minor compared to the huge sacrifice of computational efficiency brought by the exponentially growing volume of parameters.

Precision agriculture has been witnessing the increasing use of artificial intelligence. Compared with the traditional expert-involved field observations, drones or UAVs equipped with imaging sensors or multi-spectral sensors can realize field observation (e.g., pest infection monitoring) with much higher efficiency and competitive accuracy. Moreover, facilitated by deep learning techniques, artificial intelligence-based pest infection detection has shown its great potential for large-area rapid field observation. However, transferring the current CNN models for plant disease detection encounters a major challenge in real scenarios. Specifically, on one hand, those heavy models running on large computation resources are costly in both economics and labor for software management and hardware maintenance. On the other hand, when utilized for large-area field observation, plant disease detection from a large number of high-resolution plant images would take a very long time. Take UAV-based imaging observation as an example, assuming one image is sampled per square meter, fields with hundreds of square kilometer areas have hundreds of million images that would take several days for the current models to analyze. Such a long analysis time is not acceptable considering rapid disease spread within several days. Though the above-mentioned lightweight models such as MobileNet and Xception have been proposed for fast-speed image classification, they were intentionally designed to deal with natural images (e.g., cars, human faces, and animals) instead of plant leaf images. Existing model architectures should be optimized to accelerate model inference speed further while maintaining high accuracy on plant images. Accordingly, in this paper, we propose an ultra-lightweight efficient network (ULEN) aiming to strike an excellent balance between detection performance, time efficiency and computation cost, specifically for the task of image-based plant disease and pest infection detection.

Materials and methods

Datasets

The publicly available Plantvillage dataset (Hughes and Salathé, 2015) is applied in this work for experiments. The dataset consists of 54,306 images covering healthy and diseased or pest-infected leaves of 14 plants. Each image is attached with a label pair of plant type and disease type. Each plant image set includes at least 1 disease-infected image subset. There are a total of 38 classes in the dataset. Figure 1 presents some sample images of the dataset. It can be seen from Fig. 1 that image brightness and backgrounds vary in the

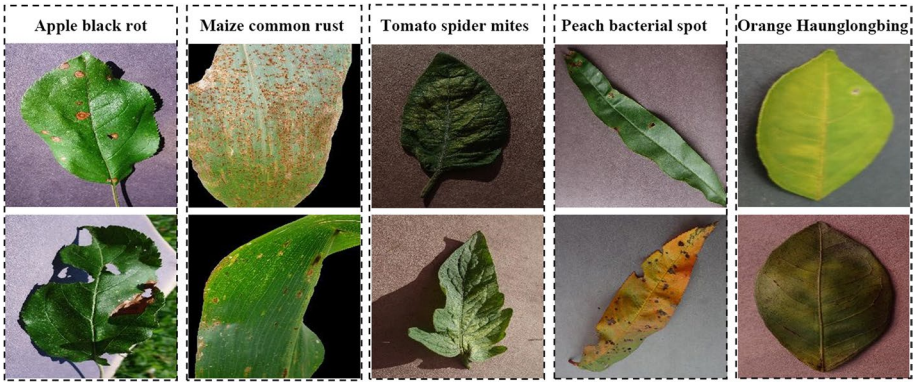


Fig. 1 Example images of Plantvillage dataset

same disease category, which brings challenges for image classification. Such variability in inner-class images enhances model generalizability, which guarantees classification confidence on images taken under different environments (e.g., shooting angles, light exposures, light contrasts).

However, it should be noted some imaging variabilities in inter-class images of this dataset pose some concerns for practical usage. Specifically, as shown in Fig. 2, grape black measles images show specular reflection while grape leaf blight images show little specular reflection. Such variability might mislead the network to a short-cut solution which is to classify grape images with specular reflections into the black measles category and classify those without specular reflections into the leaf blight category. Therefore, it should be mentioned that results are to be taken with caution for practical usage and may be influenced by unknown sources of variability between classes.

Table 1 lists the 18 plants and the corresponding diseases. Image counts of each subcategory (i.e., disease type) vary a lot. For example, the category of yellow leaf curl virus

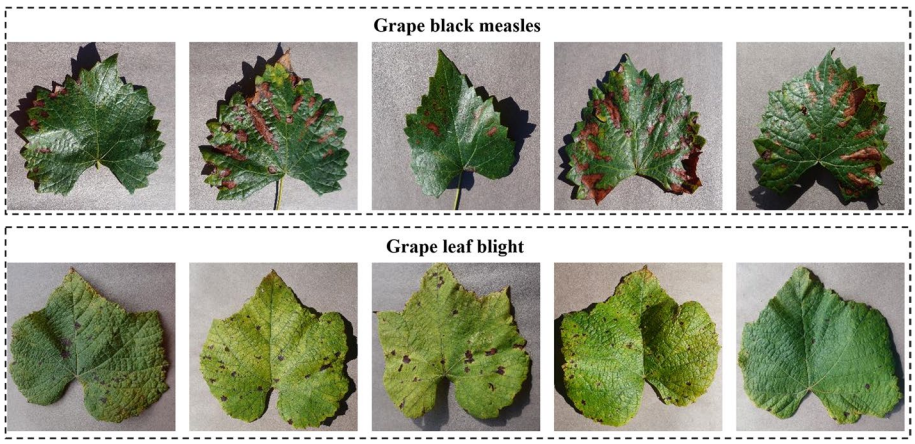


Fig. 2 Examples of imaging variability across inter-class images. Grape black measles images show specular reflection while grape leaf blight images show little specular reflection

Table 1 Statistics of Plantvillage dataset

Plant classes	Disease classes	Number of images	Total number of images
Apple	Healthy	1645	3171
	Scab	630	
	Black rot	621	
	Cedar apple rust	275	
Blueberry	Healthy	1502	1502
Cherry	Healthy	854	1906
	Powdery mildew	1052	
Maize	Healthy	1162	3852
	Gray leaf spot	513	
	Common rust	1192	
	Northern leaf blight	985	
Tomato	Healthy	1591	18 160
	Bacterial spot	2127	
	Early blight	1000	
	Late blight	1909	
	Leaf Mold	952	
	Septoria leaf spot	1771	
	Spider mites	1676	
	Target Spot	1404	
	Tomato mosaic virus	373	
	Tomato yellow leaf curl virus	5357	
Squash	Powdery mildew	1835	1835
Strawberry	Healthy	456	1565
	Leaf scorch	1109	
Soybean	Healthy	5090	5090
Potato	Healthy	152	2152
	Early blight	1000	
	Late blight	1000	
Grape	Healthy	423	4062
	Black rot	1180	
	Black Measles	1383	
	Leaf blight	1076	
Orange	Huanglongbing	5507	5507
Peach	Healthy	2297	2657
	Bacterial spot	360	
Pepper bell	Healthy	1478	2475
	Bacterial spot	997	
Raspberry	Healthy	371	371

disease in tomatoes has 5357 images accounting for 9.86% of the whole dataset. While the category of cedar apple rust has 275 images accounting for only 0.5% of the whole dataset.

The Cassava dataset

The Plantvillage dataset is the largest plant disease dataset taken with consistent image backgrounds under the indoor environment. However, the Plantvillage dataset fails to present field observed plant images in real-world scenarios in two aspects: (1) *Mixed backgrounds*. Various backgrounds (e.g., farmlands) usually exist in the images. (2) *Occlusion and overlapping of leaves*. Because deep learning models tend to focus on the most discriminative lesions, mixed backgrounds introduce noise features that may confuse the models. The obscured lesions caused by the occlusion and overlapping of leavesbring difficulty for accurate detection. Therefore, the Cassava leaf disease dataset [Mwebaze et al., 2019] containing 21,397 images collected in Uganda was used to test model performances under real-world scenarios. Most Cassava images in the dataset were taken by farmers in their gardens and annotated by experts at National Crops Resources Research Institute (NaCRRI) and Makerere Artificial Intelligence Lab. The images were labeled into the four most common cassava disease categories and healthy cassava images. The four cassava disease categories are cassava brown streak disease, cassava mosaic disease, cassava bacterial blight, and cassava green mite. As shown in Fig. 3, different Cassava disease images show very similar symptoms. Moreover, the mixed image backgrounds, the varying light conditions, and the occlusion of leaves in the images posed new challenges for the Cassava disease classification task. Same as the Plantvillage dataset, the Cassava dataset was split into three subsets with percentages of 70%, 20%, and 10% for training, validation, and test purposes, respectively. All the images were resized into 256×256 pixels to fit in model inputs.

It should be noted that the Cassava dataset also sufferd from the category imbalance problem as cassavas are more vulnerable to some specific diseases (e.g., cassava mosaic

Fig. 3 Example images of Cassava diseases

Table 2 Statistics of the Cassava dataset

Disease classes	Number of images
Cassava bacterial blight	1087
Cassava brown streak disease	2189
Cassava green mite	2386
Cassava mosaic disease	13 158
Healthy	2577

disease). Table 2 gives the number of each category. The cassava mosaic disease images account for the majority of the dataset (61.49%). The cassava brown streak disease, the cassava green mite, and the healthy images account for a similar percentage of the dataset (all in about 10%). Cassava bacterial blight has the least amount, which makes the classification of this disease more difficult.

Ultra-lightweight efficient network architecture

The basic backbone structures of VGGNet and AlexNet are both constructed by several sequentially connected conv blocks, as shown in Fig. 4a. Though the design of shared weights in the convolutional layer decreases the number of network parameters to a certain amount, network volumes grow rapidly with the network going 'deeper'. Therefore, naively stacking large convolution operations is computationally expensive. Rather than

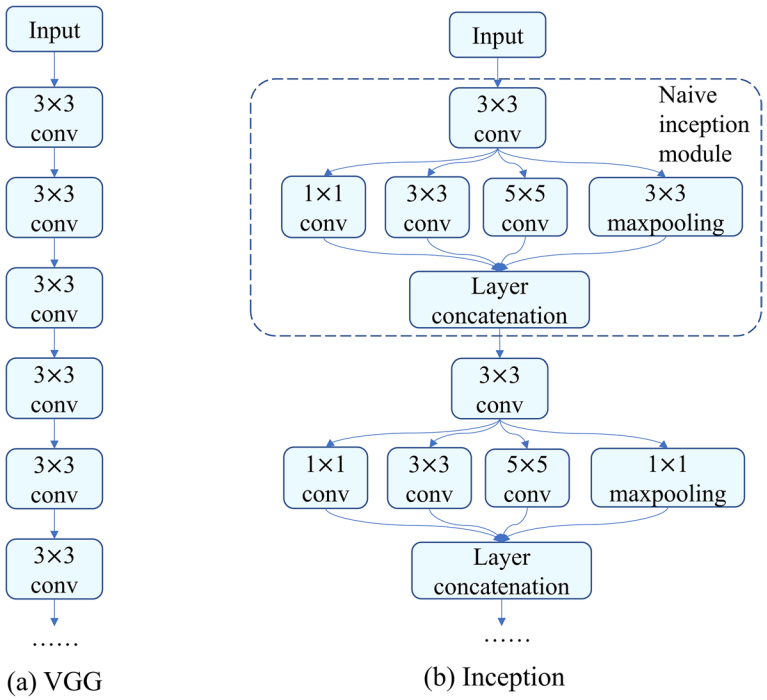


Fig. 4 Network architecture of **a** VGG network, and **b** Inception network

going 'deeper', the Inception network goes 'wider' by putting filters with multiple sizes on the same level feature maps as shown in Fig. 4b. It performs convolution on input with 3 different sizes of filter (1×1, 3×3, 5×5). A 3×3 maxpooling layer is also performed in the inception module. Outputs of the four operations are concatenated and sent to the next inception module. Since existing state-of-the-art classification networks were originally constructed for natural images of which details are very fine grained, it is necessary to expand the depth and width of networks to extract more high-level information. However, image-based plant disease classification is comparatively simpler due to the structured texture and limited color representation of plant leaves. Utilizing a very deep network for the task encountered the overfitting problem (Shi et al., 2017). Additionally, either stacking more convolutional layers vertically (i.e., going 'deeper') or horizontally (i.e., going 'wider') brings more computation burden.

In this work, a high performance while computational efficient network ULEN for image-based plant disease and pest infection detection is proposed considering the specialty of leaf image-based classification task. As shown in Fig. 5, ULEN consists of five sequentially stacked blocks. Leaf images with fixed sizes are first input to the first convolution block to extract low-level features (e.g., leaf edge features). A basic convolution block contains a convolutional layer that slides over the feature map horizontally and vertically, a pooling layer to reduce spatial dimensions of extracted feature maps, and a batch normalization layer to train network faster and more stably by normalizing the layer inputs across the batch dimension. Then two max pooling layers are used to reduce the spatial dimensions of feature maps. Reduced feature maps are then sequentially fed into three residual depth-wise convolution (RDWConv) blocks and a 1×1 convolution block to further extract high-level features embedded in low-level features. Finally, converted by spatial pyramid pooling (SPP), high-level features are flattened and are fed into a linear classification to realize image classification. Details of RDWConv and SPP are detailed in the following section.

Residual depth-wise convolution

As shown in Fig. 6 a, a standard convolution layer uses a 3-d filter to extract higher level feature maps. By sliding each 3-d filter over each pixel, one single feature map can be acquired. Therefore, the total number of trainable parameters of standard convolution is calculated as:$inputchannels \times height \times width \times outputchannels$. For example, given an input channel of 16 and output channel of 16 with 3×3 filters, the total number of trainable parameters is 2304. Accordingly, except for the first and last convolutional layer, the depth-wise convolution layer in ULEN is introduced. Instead of using a 3-d filter sliding over all input channels, depth-wise convolution uses multiple 2-d filters sliding over

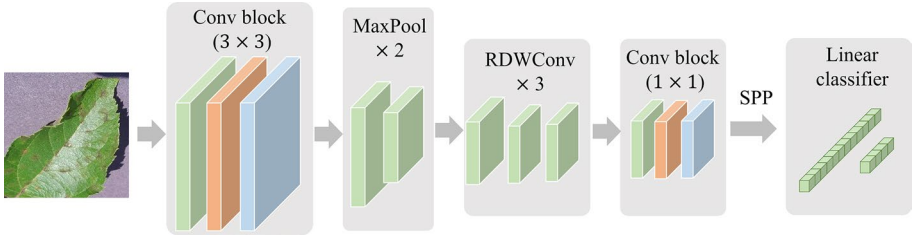


Fig. 5 Architecture of the proposed ULEN

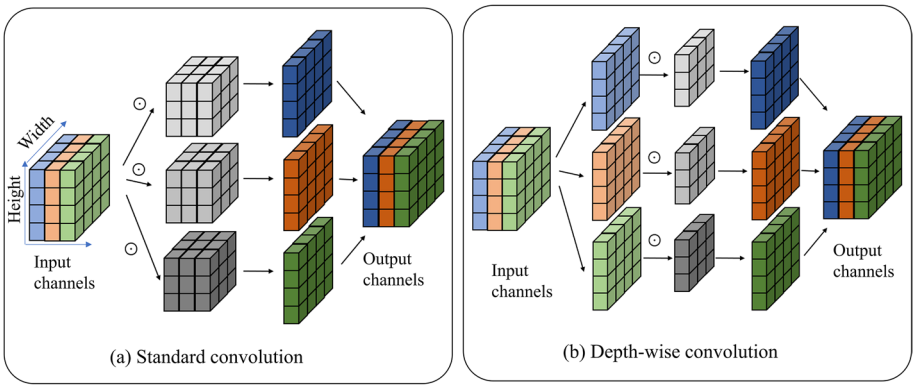


Fig. 6 Depth-wise convolution

each individual channel. For example, as shown in Fig. 6(b), the three channel input feature maps are firstly split into three independent images, then each image is convolved by the corresponding 2-d filter to get the corresponding output. Finally, the computed three outputs are stacked together to get the final output. Number of trainable parameters are $inputchannels \times height \times width$. In this way, given the above scenario, the number of trainable parameters is dramatically dropped from 2304 to 144, which gains 6× computational efficiency improvement.

Depth-wise convolution operates only on a spatial dimension for each individual channel map. To further improve information exchange across different channels, a 1×1 standard convolution layer is attached after each depth-wise convolution layer. Finally, a basic RDWConv block is constructed as shown in Fig. 7a by integrating the design of residual network (Zhang et al., 2017). By bridging the downstream layers with upstream layers using skip connections, residual can effectively avoid the vanishing gradients problem. In terms of feature map downsampling, the standard max pooling layer and average pooling layer are mostly used and are very efficient with little computational cost. However, it directly discards about one third of the embedded information. To downsample the feature map while avoiding information loss, the RDWConv block with convolution stride equal to 2 is further proposed (as shown in Fig. 7b).

Spatial pyramid pooling

After acquiring the final deep features from the last convolutional layer, a linear classification model is used to categorize deep features. Since the linear classification model accepts only 1-d feature inputs, 2-d feature maps need to be converted into 1-d feature arrays. A direct way is to reshape features by flattening. However, this way introduces too many parameters, which makes it hard for model training. For example, flattening a $16 \times 16 \times 128$ feature output will get 32,768 parameters. Another way is to use global pooling for each channel map. Therefore, flattening a $16 \times 16 \times 128$ feature output will get 128 parameters. Though global pooling can dramatically reduce model sizes, local disease-related features distributed in image corners (e.g., rust spots on leaf edges) are ignored, thus degrading model performances. Accordingly, SPP for efficient feature flattening in ULEN is introduced. As shown in Fig. 8, instead of operating only one global

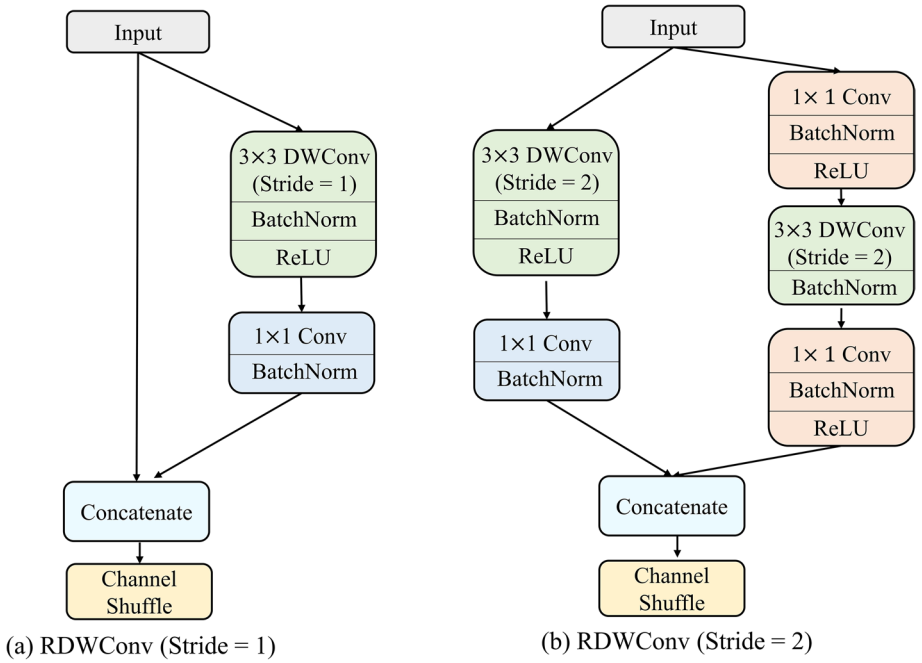


Fig. 7 Residual depth-wise convolution block

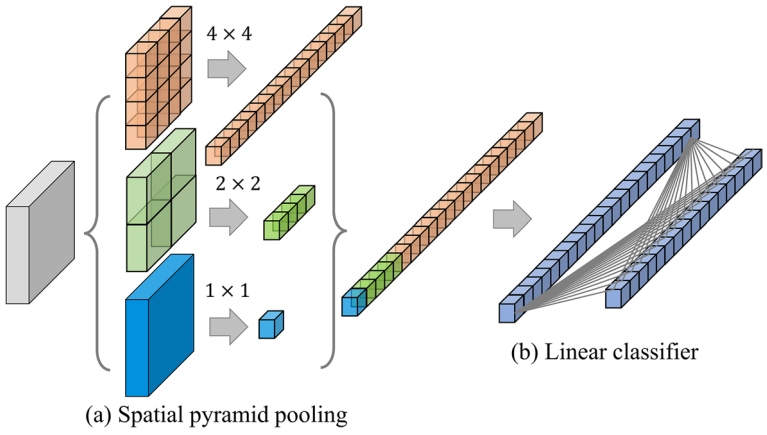


Fig. 8 Architecture of spatial pyramid pooling layer

pooling on the whole feature map, SPP utilizes a couple of different output sized pooling operations and combines the results before sending to the linear classifier. On the one hand, the sparse local features conveying disease-related information and the global features conveying plant-related information can be effectively aggregated. On the other hand, model volume can be significantly reduced.

Experiments

Model training

Experiments of ULEN and four benchmark methods were implemented in PyTorch library. Adam optimizer was utilized for model training. Batch size was set to 16. Learning rate was initially set to 0.001 for fast model learning in the first 5 epochs and then was decreased to 0.0001 to slowly converge to its best performance. Tested on the Plantvillage dataset, the models converged to a steady performance after 20 epochs. Therefore, training epoch of each model was set to 20 for model training. Cross entropy loss (CEL) was used to calculate the gaps between the predicted results and image labels. CEL is defined as follows:

$$CEL = -\frac{1}{N}\sum_{i=1}^{N} y_i \times \log\left(p_{y_i}\right) + (1 - y_i)\log(1 - p_{y_i}) \tag{1}$$

where y_i represents ground truth labels of each image, N represents the number of images in each epoch and p_{y_i} represents the predicted probability. Model parameters and training hypermeters were fine tuned and set to the same on ULEN and four benchmark methods for a fair comparison. Generally, the whole dataset was randomly split into three subsets for training, validation and test purposes in a certain percentage. Since the Plantvillage dataset has a severe data imbalance problem, random selection from the whole dataset would cause images of small categories excluded from the training dataset. Therefore, data split was conducted on each sub-category to make sure each category is included in the three datasets.

Performance evaluation

To comprehensively evaluate the performances of ULEN and the other five benchmark methods from aspects of classification accuracy and computational efficiency. The adopted metrics are introduced in this section. Three classification accuracy metrics are defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{4}$$

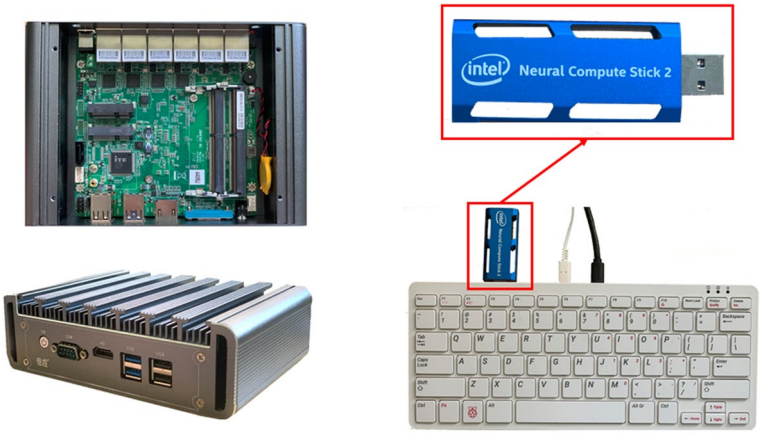
where TP, FP, TN, and FN represent true positives, false positives, true negatives and false negatives, respectively. It should be noted that for the multi-category classification tasks, two F1 scores can be utilized: micro F1 score and macro F1 score. Macro F1 is an arithmetic mean of the per-class F1-scores, while micro F1 firstly computes precision and recall on all the samples and then computes the F1 score. Here the macro F1 score was used to fairly evaluate model performance on all categories. Additionally, a confusion matrix is

introduced for visual comparison of classification accuracy. The elements on the diagonal in the confusion matrix represent the correctly classified instances along all categories. The elements of the non-diagonal represent the falsely classified instances.

In terms of computational efficiency, three metrics were measured in the experiments: Floating point operations (FLOPs), total parameter and total memory usage. Specifically, FLOPs was used to describe how many operations (such as add, subtract, multiply, and divide) were required to run a single instance of a given model. Total parameter is the sum of parameters in all layers of CNNs. Besides the network parameters, during the process of backward propagation, the activations and the associated gradients for each neuron needs to be saved. Therefore, the total memory usage of each model was further evaluated to test their applicability on devices with limited memory spaces.

Computational efficiency comparison on compact devices

To further evaluate model performance in practical scenarios, especially on devices with lower computation ability, the proposed ULEN and the other benchmark methods were tested on two compact devices. As shown in Fig. 9, a mini-PC and a Raspberry Pi4 that were used in the experiments are of very compact size. The mini-PC has an ultra-low voltage dual-core CPU (Intel Pentium 4405U, 2.1 GHz) with 8GB of memory. The Raspberry Pi4 has an even lower computational ability with a 1.8 GHz CPU (Broadcom BCM2711) with 4GB of memory. Both devices are at affordable prices (The mini-PC costs about 150 USD and the Raspberry Pi4 cost about 50 USD) and are easily deployed in any situation with little maintenance costs. It should be noted that since Raspberry Pi4 has a very limited model inference ability, an Intel neural compute stick was integrated with it to dramatically accelerate its inference time at an acceptable cost (an Intel neural compute stick 2 costs about 150 USD). Mini-PC runs on Ubuntu 22 and Raspberry Pi 4 runs on Raspberry Pi OS. Python-based deep learning toolkits Pytorch and OpenVINO were installed on the two devices, respectively. Compared with large PCs with high-performance graphic



(a) Mini-PC (b) Raspberry Pi4

Fig. 9 Two compact devices used for performance testing. **a** Mini-PC. **b** Raspberry Pi4 is equipped with an intel neural compute stick2

computation cards, the two devices have dominant superiorities over accessibility, portability, reliability and maintainability, which are very favorable for practical scenarios.

Results and discussion

Training curves

To monitor and compare the converge speeds and the stabilities of CNNs during the training phase, training loss curves and F1 score curves of the five models are shown in Figs. 10 and 11. It can be observed from Fig. 10 that all the models achieve high F1 scores on the validation dataset from 0.91 to 0.99. MobileNet_V2, Xception, ShuffleNet_V2, and ULEN quickly reach their peak performances and remain stable on the validation dataset after 6 training iterations. VGG16 converged at the lowest speed. Similarly, in Fig. 11, training losses of MobileNet_V2, Xception, ShuffleNet_V2, and ULEN reached their lowest values quickly after 6 training iterations, and difficult to further improve model performance. However, the loss curve in VGG16 fluctuated widely even when approaching 20 training epochs, which implies the instability of VGG16.

Classification accuracy evaluation

Table 3 shows the classification performance of the proposed method compared to the other five benchmark methods. It can be seen from the table that all models achieved satisfactory results, with F1 scores above 0.90. Specifically, Xception outperformed all the other methods with the highest precision (99.23%), recall (99%), and F1 score (0.991). Compared with Xception, ULEN achieved the second-best classification performance with slightly decreased precision (1.1%), recall (1.51%), and F1 score (0.0134).

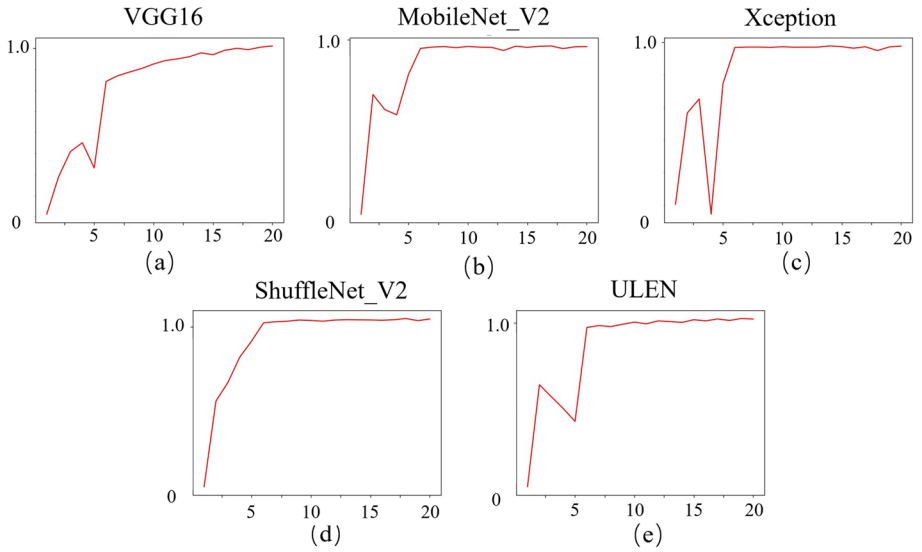


Fig. 10 F1 score curves of **a** VGG16, **b** MobileNet_V2, **c** Xception, **d** ShuffleNet_V2, **e** ULEN

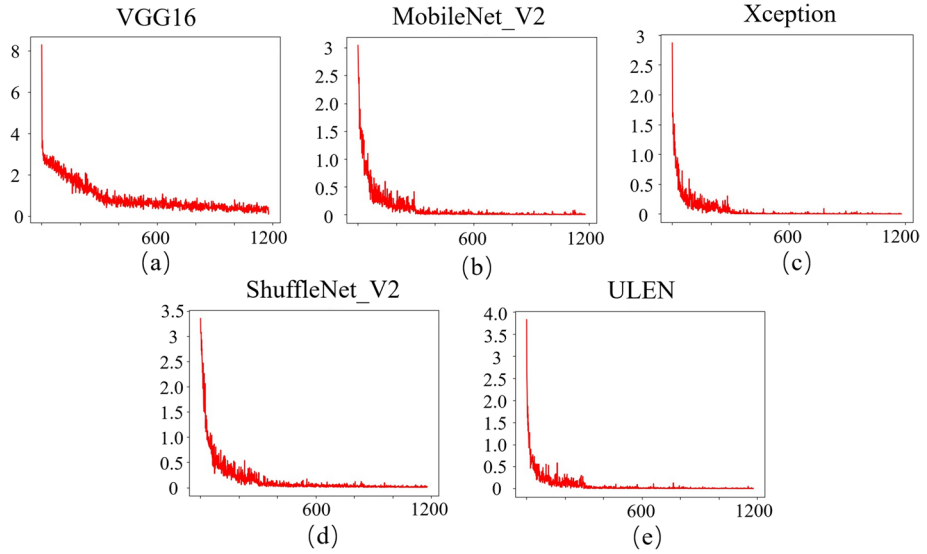


Fig. 11 Loss curves of **a** VGG16, **b** MobileNet_V2, **c** Xception, **d** ShuffleNet_V2, **e** ULEN

Table 3 Classification performances of ULEN and four benchmark models on the Plantvillage dataset

Deep learning model	Precision	Recall	F1 score
VGG16	0.9179	0.9201	0.9177
MobileNet_v2	0.9762	0.9775	0.9767
Xception	**0.9923**	**0.9900**	**0.9910**
ShuffleNet_v2	0.9721	0.9681	0.9698
ULEN	0.9813	0.9749	0.9776

Bold values indicate best performance

MobileNet_v2 achieved a very close performance compared to ULEN, with a minor performance drop on the F1 score (0.9767 compared to 0.9776). ShuffleNet_v2 ranked fourth in terms of classification accuracy, with precision (0.9721), recall (0.9681), and F1 score (0.9698). VGG16 performed the worst among all methods with significantly reduced precision (6.34% compared to ULEN), recall (5.48% compared to ULEN) and F1 score (0.0599).

To visually check the classification results, confusion matrices of results predicted by the five methods are shown in Figs. 12, 13, 14, 15 and 16. Orange grids on the diagonal of the confusion matrix represent correctly classified instances. Blue and green grids on the non-diagonal of the confusion matrix represent the number of misclassified instances. The darker the color, the more misclassified the instances were. The X-axis represents predicted categories. The Y-axis represents true labels. All the axis labels are in short abbreviations considering the limited figure space. As seen in Fig. 12 and 22 Maize_GLS (Gray leaf spot) images were misclassified as Maize_NLB (Northern leaf blight) categories, which dominates the bias of results produced by VGG16. Compared with other plant diseases, tomato diseases are more easily misclassified. Specifically,

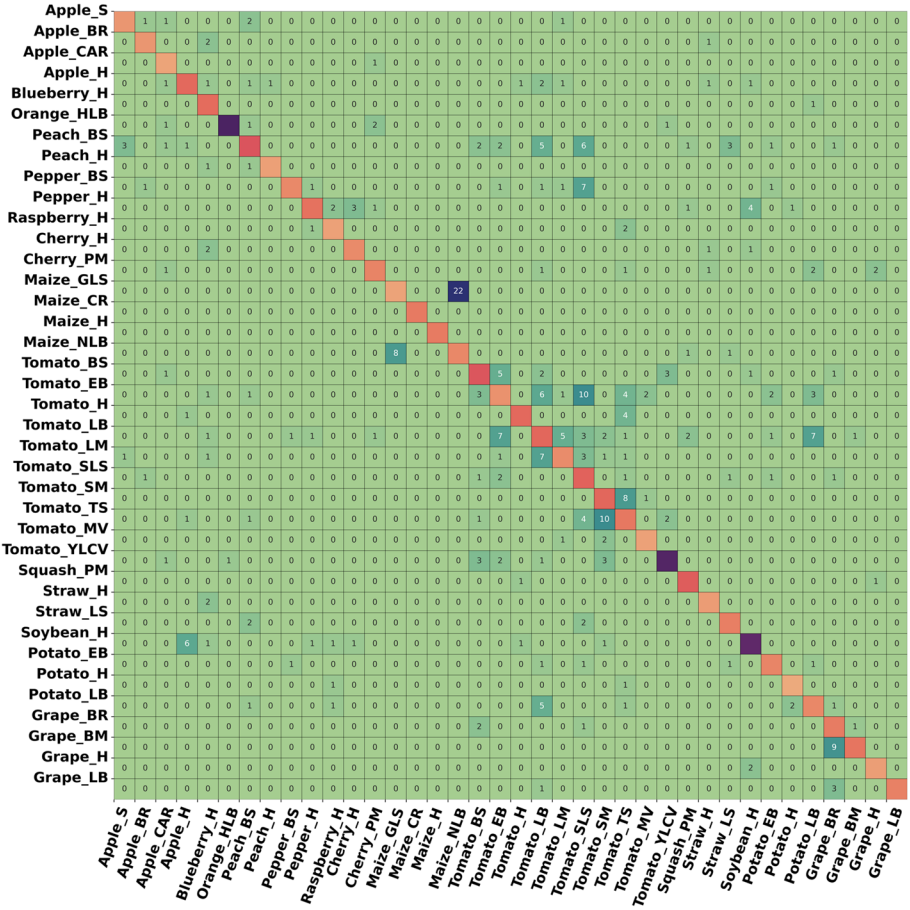


Fig. 12 Confusion matrix of results by VGG16

10 Tomato_EB (Early blight) images were misclassified as Tomato_SLS (Septoria leaf spot) disease. 10 Tomato_TS (Target spot) were misclassified as Tomato_SM (Spider mites) disease. Similar classification biases among tomato diseases can also be observed in Figs. 13, 14, 15 and 16.

The top 3 misclassified categories are presented in Figs. 17, 18 and 19 to further detail the classification biases. Specifically, Figs. 17 and 18 show results of similar diseases on the same plants. Figure 19 shows examples of confused predictions among different plant species. As shown in Fig. 17, the four tomato late blight images on the right side are falsely classified into the tomato early blight category. Tomato leaves that are infected by early blight and late blight diseases both show symptoms of dark irregular-shaped lesions. The highly similar leaf textures and colors between the two tomato diseases may confuse the current deep learning models.

Figure 18 shows some examples of confused classified images regarding Maize diseases. As shown in the figure, maize leaves infected by the gray leaf spot disease have small oval lesions with light-tan color. With the development of the disease, lesions expand into long, narrow, rectangular lesions parallel to the leaf veins. Meanwhile, the maize northern

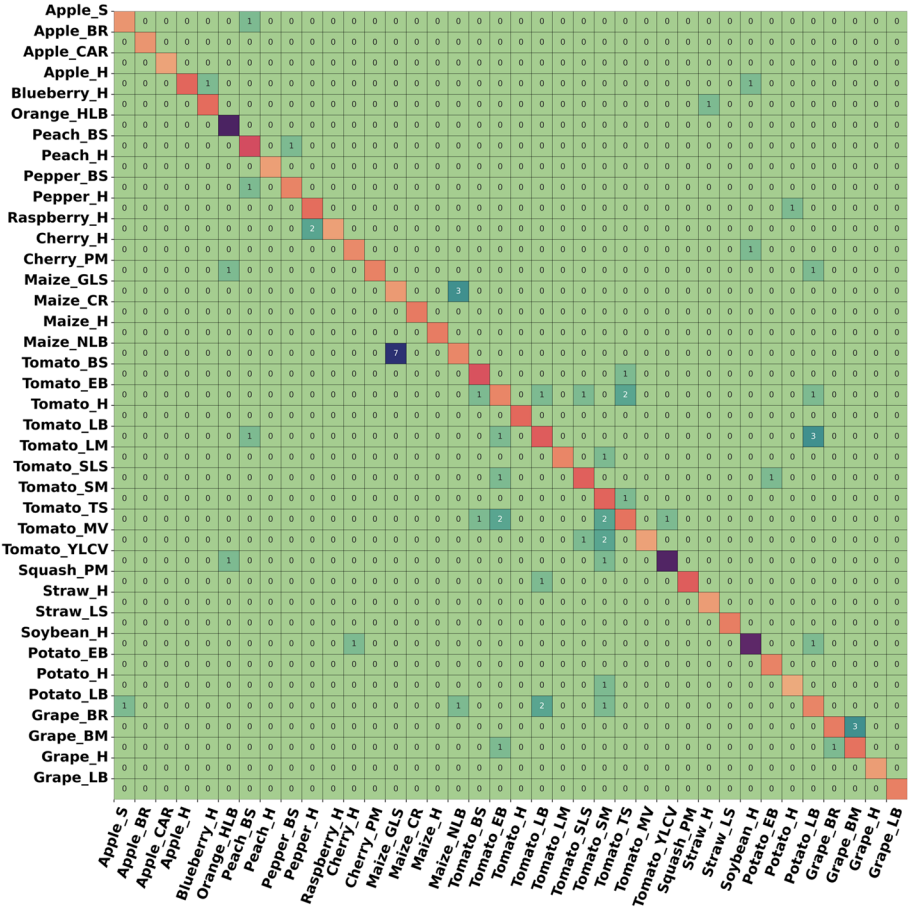


Fig. 13 Confusion matrix of results by MobileNet_v2

leaf blight disease also shows cigar or elliptical-shaped lesions that are longitudinally distributed with leaf veins, which makes the maize gray leaf spot disease easily confused with the maize northern leaf blight disease.

Figure 19 presents examples of confused predictions across different plant species. The four potato late blight images on the left side of the figure show huge different symptoms from each other. For example, the top left image shows a very small oval lesion on the leaf while the bottom left one shows a large rotten lesion in irregular shapes. The different symptoms can not only be observed from the leaf textures, but also from the leaf colors. Such discrepancies make deep learning model hard to learn a uniform representation to successfully discriminate the true potato late blight disease. To conclude, though deep learning models show very high precisions on most plant disease cases, their performance is still hindered by the limited reception field neglecting the contextual information. Expert knowledge needs to be incorporated to fully discard prediction biases.

The Cassava dataset is more challenging compared to the Plantvillage dataset due to the complex image backgrounds and the varying imaging qualities, which results in dramatic performance drops for ULEN and the other four benchmark methods. As shown in

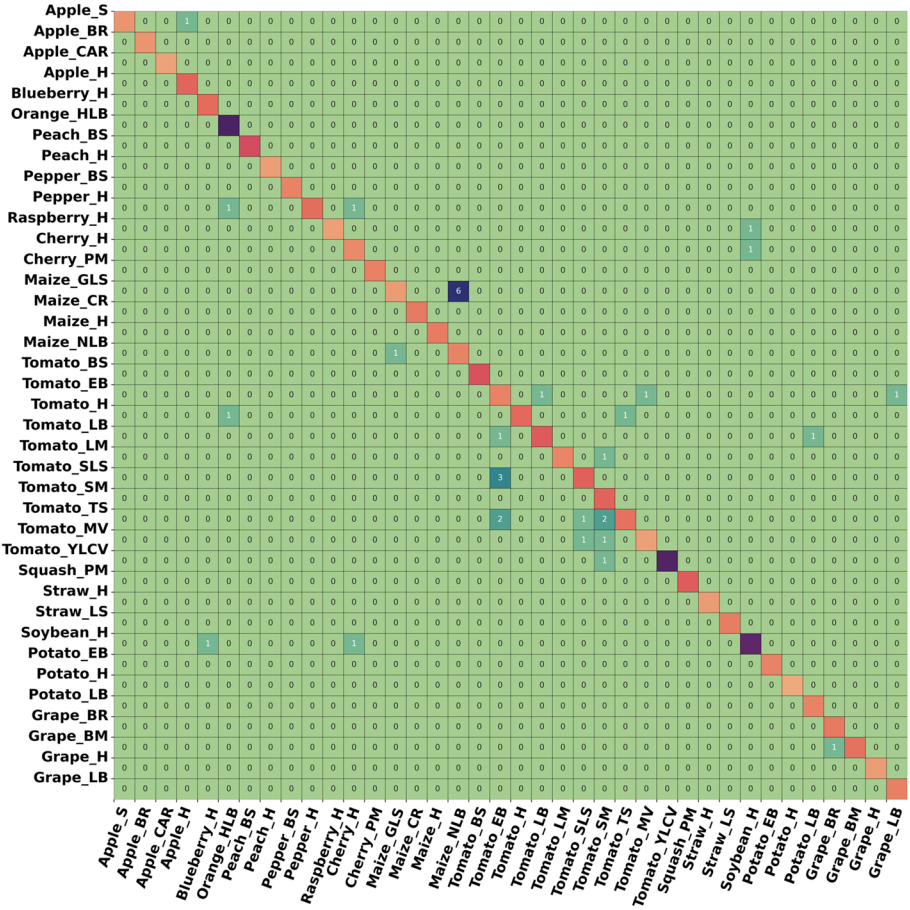


Fig. 14 Confusion matrix of results by Xception

Table 4, Xception again achieved the best performance while the precision, recall, and F1 score were very low (i.e., 68.61%, 68.6%, and 0.6843). VGG16 achieved a similar performance with Xception with an F1 score of 0.6507. ULEN achieved an F1 score of 0.5337, outperforming MobileNet_v2 and ShuffleNet_v2 with 0.0049 and 0.0582 improvement, respectively.

Computation efficiency evaluation

The computation efficiency of the proposed architecture with four benchmark methods were compared from three aspects: Total parameter, total memory and floating-point operations per second (FLOPs). As shown in Table 5, ULEN has significantly fewer parameters compared to the other four architectures. The total parameters of ULEN are 111,758, which is nearly 1484 times lighter than VGG16, 200 times lighter than Xception, 20 times lighter than MbileNet_v2, and 3.4 times lighter than ShuffleNet_v2. In terms of total memory, ULEN occupies the least memory space which is 6.09 Mb. Comparatively, the total memory usage of the state-of-the-art lightweight network ShuffleNet_v2 is two times larger

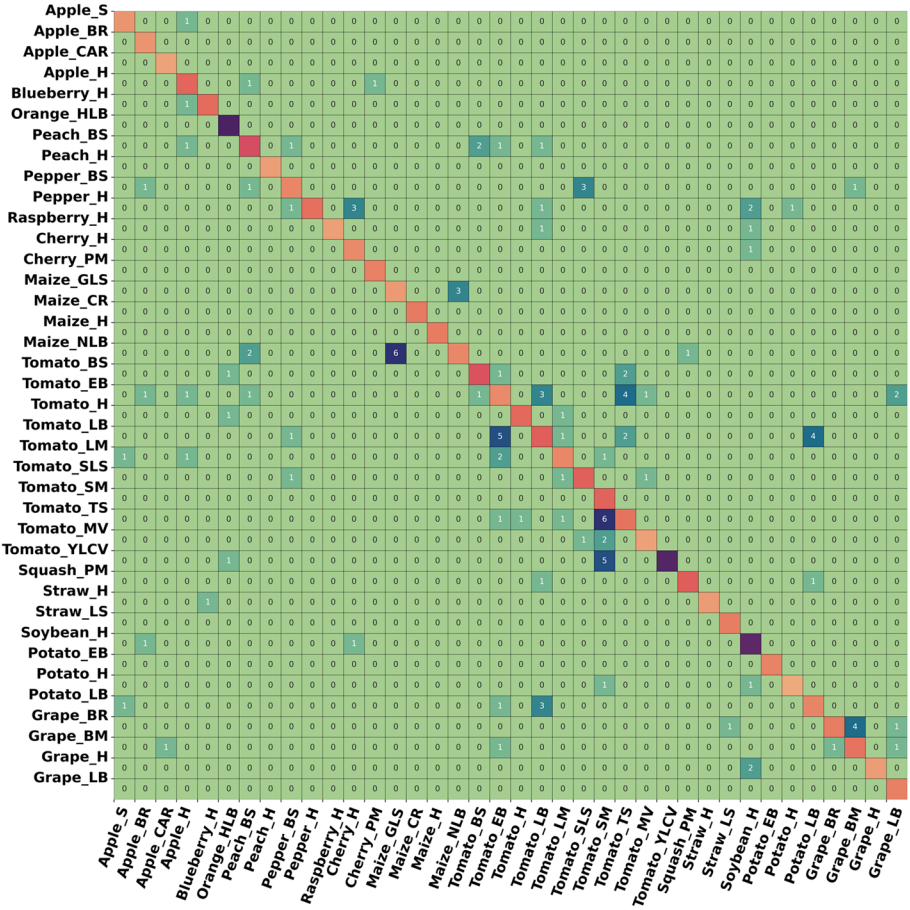


Fig. 15 Confusion matrix of results by ShuffleNet_v2

than the proposed architecture ULEN. Though Xception performs slightly better than ULEN in terms of classification accuracy, it requires a large memory usage of 190.76 Mb, which is nearly 13 times larger than ULEN. Finally, the FLOPS of the proposed method with four benchmark methods were compared. VGG16 has the largest number of FLOPs (i.e., 20.27 GFLOPs) due to its tremendous number of parameters. Xception reduces the FLOPs to 6.0 GFLOPs while reaching the highest classification accuracy. MobileNet_v2 and ShuffleNet_v2 significantly reduce the needed Flops from gigaFlops level to mega-Flops level. In the proposed architecture, ULEN has only 21.36 MFlops that run 2.6 times faster than ShuffleNet_v2 and nearly 380 times faster than Xception. From all measurements including total parameters, total memory, and Flops, ULEN dominates all the other methods. With only 111,758 network parameters and a memory requirement of only 6 Mb, ULEN has great feasibility for mobile platform deployment. The small computation power requirement (i.e., 21.36 MFlops) makes the inferencing of ULEN at a very fast speed, even running on a lightweight mobile platform.

Table 5 shows different model performances in terms of network structures. To further understand model performance in real scenarios, the proposed ULEN and the other four

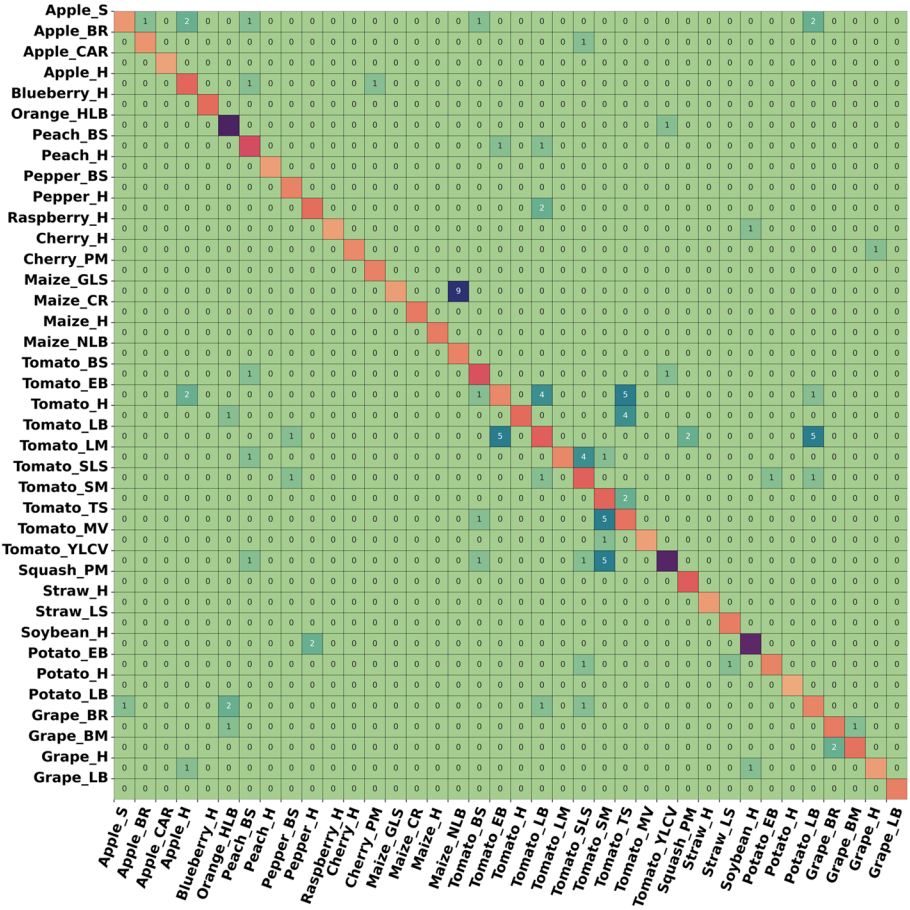


Fig. 16 Confusion matrix of results by ULEN

benchmark methods were tested on a Mini-PC and a Raspberry Pi4 to quantitatively evaluate their time efficiencies and model weight sizes. Since the process of model training relies on huge computation resources, it is typically performed on high specification PCs with high-performance GPUs. As for experiments on Mini-PC, the time required for model training on one image concerning each network was tested. Then per-image inference time of each model was tested. Moreover, to clearly show the model complexity of each network, model weight sizes are presented. As shown in Table 6, ULEN outrun all the other four methods with the least training time (0.101s), the least inference time (0.037s) and the smallest model weight size (0.5 MB). Compared to Xception which achieved the highest classification precision, ULEN run 30 times and 18 times faster than it with a 174 times smaller model weight size. The time efficiency gap between ULEN and the state-of-the-art model (i.e., ShuffleNet_v2) is around 2 to 3 times on a Mini-PC.

Table 7 shows results on Raspberry Pi4. Model load time instead of model train time was tested on Raspberry Pi4 since it does not support model training. Model inference time and weight size are also presented in Table 7. Since OpenVINO runs a different model weight format (i.e., .onnx format) from Pytorch (i.e., .pth format), model weight

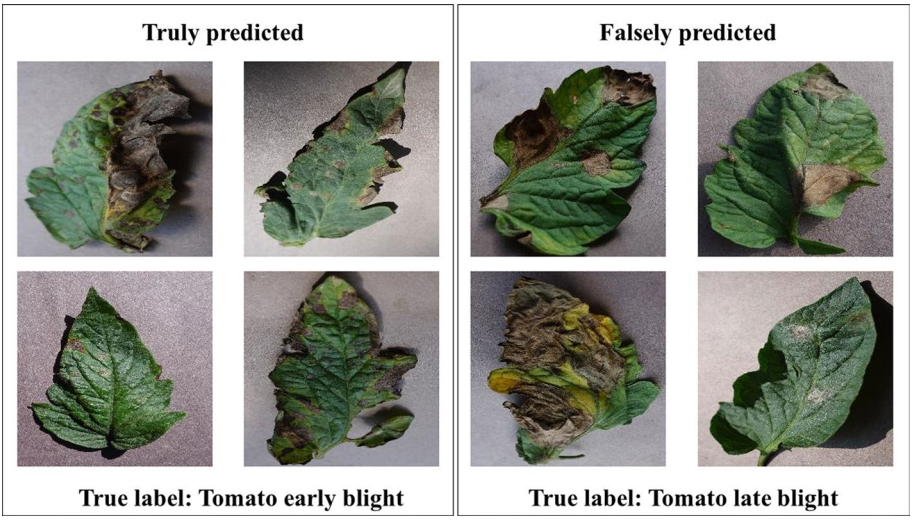


Fig. 17 The images on the left are truly predicted tomato early blight disease. The images on the right are falsely predicted

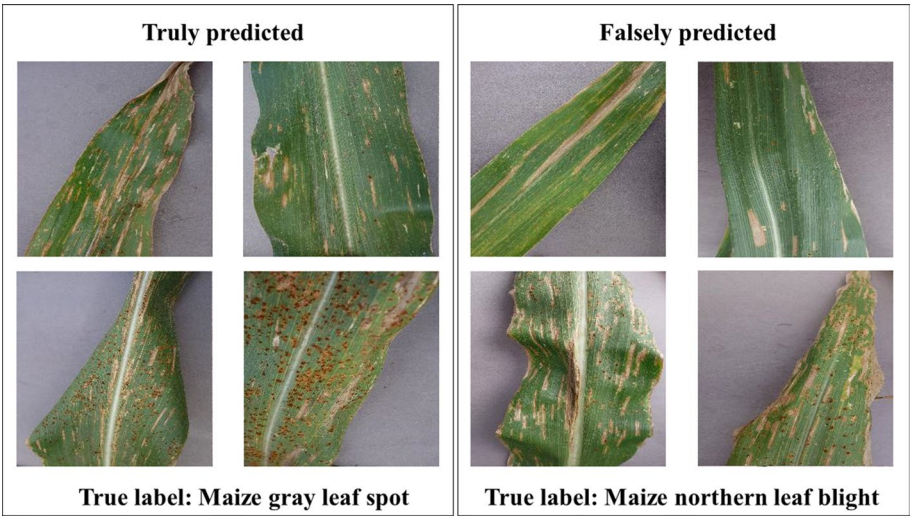


Fig. 18 The images on the left are truly predicted Maize gray leaf spot disease. The images on the right are falsely predicted

sizes on Raspberry Pi4 differ from those on Mini-PC. However, model load time among different networks differs a lot on Raspberry Pi4 than on Mini-PC due to the lower CPU. As shown in Table 7, VGG16 failed to load its weight file on Raspberry Pi4 considering its huge model weight size (i.e., 632 MB). Xception takes 24s to load model weight. Enhanced by Intel neural compute stick2, inference times of all models were accelerated while the gain on ULEN was marginal (i.e., 0.005s). State-of-the-art lightweight

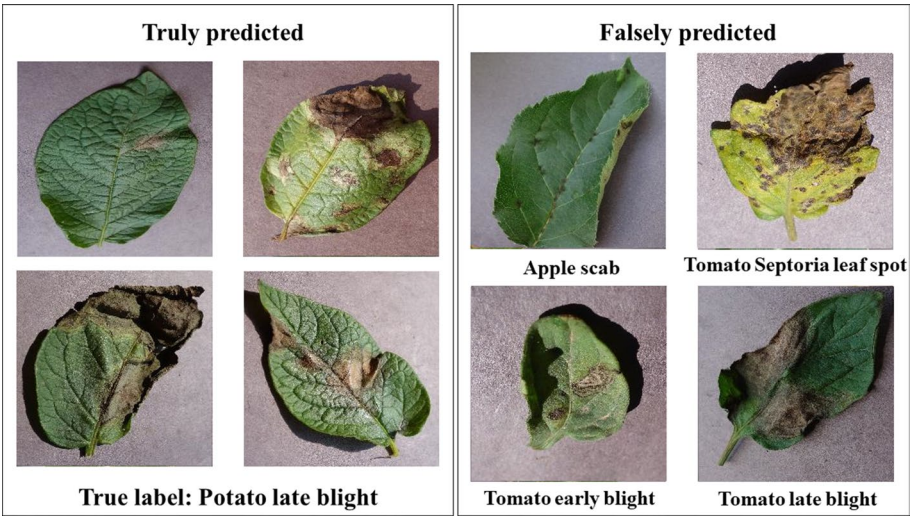


Fig. 19 The images on the left are truly predicted Potato late blight disease. The images on the right are falsely predicted

Table 4 Classification performances of ULEN and four benchmark models on the Cassava dataset

Deep learning model	Precision	Recall	F1 score
VGG16	0.6847	0.6468	0.6507
MobileNet_v2	0.5439	0.5292	0. 5288
Xception	**0.6861**	**0.6860**	**0.6843**
ShuffleNet_v2	0.5010	0.4649	0.4755
ULEN	0.5497	0.5322	0.5337

Bold values indicate best performance

Table 5 Computation efficiency of ULEN and four benchmark models. 1 M FLOPs represents 1000,000 FLOPs. 1 GFlops represents 1000,000,000 FLOPs

Deep learning model	Total parameter	Total memory	Flops
VGG16	165,873,510	210.22 Mb	20.27 GFlops
Xception	22,855,952	190.76 Mb	6.0 GFlops
MobileNet_v2	2,272,550	96.97 Mb	416.64 MFlops
ShuffleNet_v2	380,742	14.67 Mb	55.71 MFlops
ULEN	**111,758**	**6.09 Mb**	**21.36 MFlops**

Bold values indicate best performance

model ShuffleNet_v2 showed similar time efficiency on model load time and inference time. But its weight size is more than 3 times bigger than ULEN. The result suggests that the gain in inference efficiency gradually converged to a low level with the increase of the computation ability. In other words, time efficiency gaps between heavy-weight models and light-weight models were enlarged on compact devices, which highlights

Table 6 Computation efficiency of ULEN and four benchmark models on Mini-PC

Deep learning model	Train time (seconds)	Inference time (seconds)	Weight size (MB)
VGG16	10.144	1.555	632
Xception	3.102	0.671	87
MobileNet_v2	0.680	0.137	8.9
ShuffleNet_v2	0.277	0.079	1.6
ULEN	**0.101**	**0.037**	**0.5**

Bold values indicate best performance

Table 7 Performance evaluation on Raspberry Pi4 with Intel neural compute stick 2

Deep learning model	Model load time (sec)	Inference time (sec)	Weight size (MB)
VGG16	–	–	632
Xception	24.309	0.148	87
MobileNet_v2	4.501	0.055	8.6
ShuffleNet_v2	2.770	0.034	1.5
ULEN	**2.232**	**0.032**	**0.4**

Bold values indicate best performance

the necessity of ultra-light models in practical scenarios which require low computation resources.

Discussion on model performances in different scenarios

For images in the Plantvillage dataset, plant leaf photos were taken under indoor environments with identical backgrounds, and the leaves do not overlap. The main challenge lies in the discrimination of disease symptom features across different plants. Since the leaves are placed at the center of images, it is easier for conventional convolutional layers to successfully capture the deep features of different diseases even though the reception field is limited. Therefore, for models using single convolutional kernels (e.g., VGG16) or multiple convolutional kernels (e.g., Xception), satisfying performances were achieved in the high-quality Plantvillage dataset. For images taken with mixed backgrounds and occlusion of leaves in the Cassava dataset, performance gaps of different models were enlarged. The results on the Cassava dataset proved the superiority of heavy models for plant disease classification tasks. Specifically, Xception used multiple convolutional kernels with various reception fields to capture both local and global information, such that the detailed leaf features in the image center position and the background features in the image edge position are considered for disease classification. Therefore, Xception can effectively overcome the problem of mixed backgrounds and occlusion of leaves. Interestingly, VGG16 with the simplest network architecture achieved competing performances with Xception without advanced module, which demonstrates that heavy models are naturally more robust to complex backgrounds and varying imaging conditions than light models. The phenomenon is consistent with the work demonstrating that large models generalize better than small models [Brutzkus

et al., 2019]. However, their high classification accuracies were achieved at the expense of high computing loadings, which makes them hardly deployed on compact devices. A balance between classification accuracy and computation cost needs to be compromised for practical needs in real-world field observation scenarios. Following the idea of increasing the observation field, ULEN proposed to incorporate spatial pyramid pooling layers with depth-wise convolutional layers in the feature extraction network instead of using multiple convolutional kernels to avoid introducing more network parameters. As discussed in "Classification accuracy evaluation", ULEN performed inferior to the best classification performance model Xception with an F1 score drop of 0.15, but the model size is 200 times lighter than Xception. Compared with the lightest benchmark model (i.e., ShuffleNet_v2), ULEN beats it in terms of both classification accuracy and computation efficiency.

Discussion on the potential of ULEN for plant disease detection

This work focuses on an ultra-lightweight network targeting plant disease classification. This section further discusses the potential of ULEN for another highly-related task: plant disease detection. The task of plant disease detection requires not only distinguishing the type of plant disease but also specifying the exact position of the diseased part in the image. Practically, the two tasks are applied for different purposes from different perspectives. Plant disease detection fits the purpose of precise disease detection in object-level granularity while the image-level plant disease classification is more suitable for large-area rapid disease detection which does not require object-level information. Though the proposed ULEN cannot be directly applied for object-level detection, the two tasks share a core module for feature extraction, which suggests the proposed structure can be easily transferred to object detection-targeted networks. As shown in Fig. 20, typical plant disease detection networks (RCNN [Girshick et al., 2014], Mask-RCNN [Girshick 2015], Fast-RCNN [He et al., 2017]) consist of two streams that share the same feature extractor module for saving computation cost purposes. The first stream is same as the conventional image classification structure, which means the proposed ULEN structure can be directly transferred to this stream to accelerate image classification speed. The second stream re-utilizes the feature extractor in ULEN. Considering

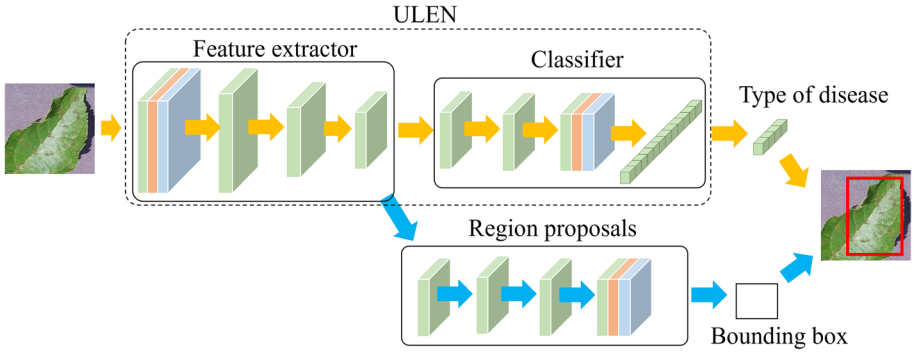


Fig. 20 A ULEN-modified object detection network for plant disease detection

the feature extractor module consumes the most computation during model training and inference, adopting ULEN for conventional plant disease detection architectures can promisingly accelerate its training and inference speed.

Conclusion

In this paper, a novel ultra-lightweight efficient network for image-based plant disease and pest infection detection is proposed. The proposed architecture consists of two parts to realize plant disease-related feature extraction and deep feature classification, by adopting residual depth-wise convolution and spatial pyramid pooling, respectively. A comprehensive comparison experiment validates the superior lightweight advantage of the proposed architecture over other state-of-the-art architectures with a slight compromise on classification accuracy. The architecture has shown a comparable classification accuracy to effectively detect plant diseases and pest infections with at least 200 times fewer parameters compared to the most accurate architecture and outperforms the state-of-the-art lightweight architecture with more than 3 times fewer parameters. Such lightweight design significantly reduces the computation power consumption of the network, which enables it to be trained on low computation power platforms or even without GPUs, thereby facilitating an environmental-friendly deployment of plant disease and pest infection detection model in practical scenarios. The proposed architecture could easily be extended to other plant (e.g., tea, tobacco) disease and pest infection detection tasks with minor adjustments.

Acknowledgements This work was supported by the Key project of Hubei Provincial Natural Science Foundation (2019CFA031), Natural Science Foundation of Hubei Province (2022CFB657), Science and technology project of Hubei Tobacco Company (027Y2021-022), Youth Science Fund Project of Hubei Academy of Agricultural Sciences (2023NKYJJ27).

Author contributions Study conception and design were proposed by BW and CZ. Material preparation and data collection were performed by YL, CC, DH, and YG. Analysis was performed by BW and CZ. The first draft of the manuscript was written by BW. Review and editing were conducted by CZ, and all authors commented on previous versions of the manuscript.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

Atila, Ü., Uçar, M., Akyol, K., & Uçar, E. (2021). Plant leaf disease classification using EfficientNet deep learning model. *Ecological Informatics*, *61*, 101182. https://doi.org/10.1016/j.ecoinf.2020.101182.

Brutzkus, A., & Globerson, A. (2019). Why do larger models generalize better? A theoretical perspective via the XOR problem. *In International Conference on Machine Learning, 97*, 822–830.

Deng, J., Dong, W., Socher, R., Li, LJ., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *In 2009 IEEE Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2009.5206848

FAO. (2021). *Scientific review of the impact of climate change on plant pests–A global challenge to prevent and mitigate plant pest risks in agriculture, forestry and ecosystems*. Rome: FAO on behalf of the IPPC Secretariat. https://doi.org/10.4060/cb4769en

Girshick, R. (2015). Fast r-cnn. *In Proceedings of the IEEE International Conference on Computer Vision*, 1440–1448.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 580–587.

Hassan, S. M., Maji, A. K., Jasiński, M., Leonowicz, Z., & Jasińska, E. (2021). Identification of Plant-Leaf Diseases using CNN and transfer-learning Approach. *Electronics*, *10*(12), 1388. https://doi.org/10.3390/electronics10121388.

He, K., Gkioxari, G., Dollár, P., Girshick, R. (2017). Mask r-cnn. *In Proceedings of the IEEE International Conference on Computer Vision*, 2961–2969.

Hlaing, C.S., & Zaw, S.M.M. (2017). Model-based statistical features for mobile phone image of tomato plant disease classification. *In 2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies*.https://doi.org/10.1109/PDCAT.2017.00044

Hughes, D., & Salathé, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *Non-peer reviewed preprint at arXiv:1511.08060*. https://doi.org/10.48550/arXiv.1511.08060

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, *147*, 70–90. https://doi.org/10.1016/j.compag.2018.02.016.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., et al. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551. https://doi.org/10.1162/neco.1989.1.4.541.

Miller, S. A., Beed, F. D., & Harmon, C. L. (2009). Plant disease diagnostic capabilities and networks. *Annual Review of Phytopathology*, *47*, 15–38. https://doi.org/10.1146/annurev-phyto-080508-081743.

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, *7*, 1419. https://doi.org/10.3389/fpls.2016.01419.

Mwebaze, E., Gebru, T., Frome, A., Nsumba, S., & J, Tusubira, J. (2019). iCassava 2019 fine-grained visual categorization challenge. *Non-peer Reviewed Preprint at arXiv:1908.02900*. https://doi.org/10.48550/arXiv.1908.02900

Priyadharshini, R. A., Arivazhagan, S., Arun, M., & Mirnalini, A. (2019). Maize leaf disease classification using deep convolutional neural networks. *Neural Computing and Applications*, *31*(12), 8887–8895. https://doi.org/10.1007/s00521-019-04228-3.

Rançon, F., Bombrun, L., Keresztes, B., & Germain, C. (2019). Comparison of SIFT encoded and deep learning features for the classification and detection of Esca disease in Bordeaux vineyards. *Remote Sensing*, *11*(1), 1. https://doi.org/10.3390/rs11010001.

Shi, H., Xu, M., & Li, R. (2017). Deep learning for household load forecasting-A novel pooling deep RNN. *IEEE Transactions on Smart Grid*, *9*(5), 5271–5280. https://doi.org/10.1109/TSG.2017.2686012.

Singh, P., Verma, A., & Alex, J. S. R. (2021). Disease and pest infection detection in coconut tree through deep learning techniques. *Computers and Electronics in Agriculture*, *182*, 105986. https://doi.org/10.1016/j.compag.2021.105986.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational Intelligence and Neuroscience*. https://doi.org/10.1155/2016/3289801

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., & Anguelov, D. (2015). Going deeper with convolutions. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2015.7298594

Thangaraj, R., Anandamurugan, S., & Kaliappan, V. K. (2021). Automated tomato leaf disease classification using transfer learning-based deep convolution neural network. *Journal of Plant Diseases and Protection*, *128*(1), 73–86. https://doi.org/10.1007/s41348-020-00403-0.

Wen, J., Shi, Y., Zhou, X., & Xue, Y. (2020). Crop disease classification on inadequate low-resolution target images. *Sensors (Basel, Switzerland)*, *20*(16), 4601. https://doi.org/10.3390/s20164601.

Zhang, C., Yue, P., Di, L., & Wu, Z. (2018). Automatic identification of center pivot irrigation systems from landsat images using convolutional neural networks. *Agriculture*, *8*(10), 147. https://doi.org/10.3390/agriculture8100147.

Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, *26*(7), 3142–3155. https://doi.org/10.1109/TIP.2017.2662206.