agronomy

Article

Prune-FSL: Pruning-Based Lightweight Few-Shot Learning for Plant Disease Identification

Wenbo Yan [1], Quan Feng [1,*], Sen Yang [1,*], Jianhua Zhang [2] and Wanxia Yang [1]

[1] College of Mechanical and Electrical Engineering, Gansu Agricultural University, Lanzhou 730070, China; yanwb@st.gsau.edu.cn (W.Y.); yangwanxia@gsau.edu.cn (W.Y.)
[2] Agricultural Information Institute, Chinese Academy of Agricultural Sciences, Beijing 100081, China; zhangjianhua@caas.cn
* Correspondence: fquan@gsau.edu.cn (Q.F.); yangsen@gsau.edu.cn (S.Y.)

Abstract: The high performance of deep learning networks relies on large datasets and powerful computational resources. However, collecting enough diseased training samples is a daunting challenge. In addition, existing few-shot learning models tend to suffer from large size, which makes their deployment on edge devices difficult. To address these issues, this study proposes a pruning-based lightweight few-shot learning (Prune-FSL) approach, which aims to utilize a very small number of labeled samples to identify unknown classes of crop diseases and achieve lightweighting of the model. First, the disease few-shot learning model was built through a metric-based meta-learning framework to address the problem of sample scarcity. Second, a slimming pruning method was used to trim the network channels by the γ coefficients of the BN layer to achieve efficient network compression. Finally, a meta-learning pruning strategy was designed to enhance the generalization ability of the model. The experimental results show that with 80% parameter reduction, the Prune-FSL method reduces the Macs computation from 3.52 G to 0.14 G, and the model achieved an accuracy of 77.97% and 90.70% in 5-way 1-shot and 5-way 5-shot, respectively. The performance of the pruned model was also compared with other representative lightweight models, yielding a result that outperforms those of five mainstream lightweight networks, such as Shufflenet. It also achieves 18-year model performance with one-fifth the number of parameters. In addition, this study demonstrated that pruning after sparse pre-training was superior to the strategy of pruning after meta-learning, and this advantage becomes more significant as the network parameters are reduced. In addition, the experiments also showed that the performance of the model decreases as the number of ways increases and increases as the number of shots increases. Overall, this study presents a few-shot learning method for crop disease recognition for edge devices. The method not only has a lower number of parameters and higher performance but also outperforms existing related studies. It provides a feasible technical route for future small-sample disease recognition under edge device conditions.

Keywords: meta-learning; metric learning; plant protection; crop disease recognition; deep learning

check for updates

Citation: Yan, W.; Feng, Q.; Yang, S.; Zhang, J.; Yang, W. Prune-FSL: Pruning-Based Lightweight Few-Shot Learning for Plant Disease Identification. *Agronomy* **2024**, *14*, 1878. https://doi.org/10.3390/agronomy14091878

Academic Editor: Paul Kwan

Received: 8 June 2024
Revised: 19 August 2024
Accepted: 20 August 2024
Published: 23 August 2024

1. Introduction

With the rapid development of modern agriculture, the impact of plant diseases on agricultural production has become increasingly serious [1–4]. According to statistics, the global economic losses caused by plant diseases amount to hundreds of billions of dollars every year. Consequently, achieving prompt and precise diagnosis of plant diseases is critically important for enhancing crop yields and minimizing economic losses. Traditional plant disease diagnosis methods mainly rely on manual observation and expert experience, and this method has problems such as high subjectivity, low efficiency, and high misdiagnosis rates. In recent years, the rapid advancement of computer vision and deep learning technology has led to significant developments. A large number of studies have applied computer vision technology to plant disease diagnosis [5–9] and achieved excellent performance. However, plant disease diagnosis models based on deep learning still have certain limitations. Firstly, deep learning models generally necessitate a substantial amount of labeled data to ensure effective and robust training, and in the field of plant disease diagnosis, acquiring a substantial volume of labeled data remains a challenging and resource-intensive endeavor. Secondly, the high computational overhead of deep learning models and the high demand for high-performance computing resources at prediction time limit the application and popularity of the models in some scenarios with tight computing resources and high real-time performance (e.g., edge devices) [10–13].

Currently, for the problem of plant disease recognition with small samples, traditional technical routes focus on data augmentation and transfer learning. Data augmentation improves the generalization ability of the model by geometrically transforming existing samples or expanding the dataset through adversarial generative networks. On the other hand, transfer learning pre-trains models on large datasets to acquire a priori knowledge. It is later fine-tuned on downstream tasks to accomplish the target task by transferring the a priori knowledge. For example, Hu [14] et al. used a method called C-DCGAN to enhance the disease spot images. This approach resulted in an average accuracy of 90.00% for tea disease recognition. Chen [15] et al. similarly tackled the challenge of insufficient samples by utilizing a Cy-cliGAN network to generate synthetic samples, which resulted in a 97.78% in the task of apple disease classification accuracy. Cap [16] et al. proposed a novel image transformation system (LeafGAN) for crop disease characterization, which has its own mechanism and outperforms CycleGAN in image generation. The method, which leverages transfer learning [17], is initially pre-trained on a source dataset to capture a generic feature representation that can be effectively transferred to other related tasks and subsequently uses a small amount of target data for network fine-tuning. For example, Zhang [6] et al. performed transfer learning on Densenet on the Plantvillage dataset to develop a model for recognizing lotus leaf-related diseases with an accuracy of 91.34%. Li [18] et al. initially pre-trained a Densenet model on the Plantvillage dataset to capture a generic feature representation and then fine-tuned it specifically on the tea dataset, achieving 92.66% accuracy in recognizing tea diseases despite the limited availability of samples. Yang [19] et al. utilized Mobilenetv2 for transfer learning and achieved 97.23% accuracy in recognizing corn diseases. These studies show that data augmentation or transfer learning mitigates the problem of insufficient samples in crop disease recognition to a certain extent and, at the same time, prompts the model to become more lightweight. However, there are still some limitations to these methods: (1) Although the models perform well on disease categories in the training samples, they often fail to extend their predictive power to untrained disease categories; (2) data augmentation and transfer learning can still face challenges in achieving a reliable level of performance when the number of available samples is extremely limited, as very sparse samples do not provide sufficient information to meet the model's training requirements.

In recent years, small-sample recognition methods based on metric meta-learning have become a new approach to solving the sample scarcity problem. The meta-learning method avoids parameter learning within a linear layer while it can be directly generalized to downstream tasks under the condition of very few samples. Currently, the meta-learning-based recognition method has good application prospects in the field of crop disease recognition. For example, Pan [17] et al. proposed a few-shot learning (FSL) method specifically for crop leaf disease recognition. On monocotyledonous crops, the method achieved 68.57% and 76.95% accuracy in 5-way 5-shot and 10-way 10-shot, respectively. Xiao [20] et al. compared the performance of different feature encoders in prototypical, matching, and relational networks, and the average accuracies of the three models on the Plantvillage dataset were 77.60%, 73.01%, and 73.13% under the 5-way 1-shot setting, respectively. These findings provide an important reference for evaluating the feasibility of FSL in the field of crop disease identification. Lin [21] et al. proposed a network architecture that combines multi-scale features with channel attention mechanisms to enhance feature representation. This method achieved excellent performance on both 5-way 1-shot and 5-way 5-shot tasks on the Plantvillage dataset. In summary, these studies demonstrate the great potential of few-shot learning (FSL) in the field of crop disease recognition. Our research also achieved high-precision identification of plant diseases by combining three few-shot learning networks. However, the large number of parameters and the computational complexity of the FSL model based on the backbone network Resnet12 [22] in the above studies limit the application of the model in resource-constrained environments, especially on mobile devices and embedded systems. This does not leverage the advantage that FSL can be deployed directly without fine-tuning. Therefore, designing a lightweight backbone for FSL that aims to significantly reduce the number of parameters and computational overhead of the model while seeking to maintain or approach the superior performance of existing models has increasingly become an important topic of research in this area.

In this study, we propose a lightweight fast-learning (FSL) method called Prune-FSL, which combines the DeepEMD [23] meta-learning framework and the slimming [24] pruning technique. Specifically, we leverage DeepEMD to deal with data scarcity and use the γ coefficient to identify and prune redundant channels in the network from the bulk normalization (BN) layer, thereby effectively reducing the network parameters.

The aim of this study is to achieve efficient lightweighting of FSL with the following contributions:

(1) Combining meta-learning and model pruning algorithms, a lightweight few-shot learning method based on pruning is proposed and used for disease recognition.

(2) A pruning strategy is designed specifically to serve meta-learning to provide better generalization performance for the pruned model.

(3) The number of parameters of the method and its accuracy were compared with lightweight models such as Densenet40 [25], Shufflenetv2 [26], EfficientNet [27], Mobilenetv2 [28], and Mobilenetv3, which proved the superiority of the method. The influence of various factors affecting FSL is also analyzed through extensive experiments and compared with related work.

2. Materials

2.1. Materials

The Plantvillage [29] dataset was selected as the experimental material for this study. The Plantvillage (PV) dataset is a publicly available large-scale plant leaf image dataset containing 50,403 images covering 14 plant species with a total of 38 disease types, as shown in Table 1. In addition, the selected hardware environment for this study carries an Intel i7-12700 processor (Manufacturer: Intel Corporation; Santa Clara, CA, USA), an Nvidia RTX 3090 graphics card (Manufacturer: Nvidia Corporation; Santa Clara, CA, USA), and 32 GB of memory. The PyTorch version is 1.3.1, and the system environment is Ubuntu 23.04.

Table 1. The 14 species and 38 categories in PV and Field-PV.

Species	Class Numbers	Class Name	Number in PV
Apple	4	Apple scab, Black rot, Cedar apple rust, Healthy	3174
Blueberry	1	Healthy	1502
Cherry	2	Healthy, Powdery mildew	1905
Corn	4	Cercospora leaf spot, Gray leaf spot, Common rust, Northern leaf blight, Healthy	3852
Grape	4	Black rot, Black measles, Healthy, Leaf blight	3862
Orange	1	Haunglongbing	5507
Peach	2	Bacterial spot, Healthy	2657
Pepper	2	Bell bacterial spot, Bell healthy	2473
Potato	3	Early blight, Healthy, Late blight	2152
Raspberry	1	Healthy	371
Soybean	1	Healthy	5089

Table 1. *Cont.*

Species	Class Numbers	Class Name	Number in PV
Squash	1	Powdert mildew	1835
Strawberry	2	Healthy, Leaf scorch	1565
Tomato	10	Bacterial spot, Early blight, Healthy, Late blight, Leaf mold, Septoria leaf spot, Spider mite, Two-spotted spider mite, Target spot, Tomato mosaic virus (TMOV), Yellow leaf curl virus (YLCV)	18,159

2.2. Problem Formulation

In the field of few-shot learning (FSL), datasets are typically divided into base classes and novel classes. Base classes [30], with a large number of samples, are mainly used in the meta-training phase of the model, which is designed to allow the model to learn how to generalize the relationships between different classes. Comparatively, novel class data, with a very small number of samples, are reserved for the meta-testing phase to evaluate how well the model learns on novel classes that have not been seen before. This setup ensures complete separation of training and testing data, i.e., there is no intersection between the base class and the novel class ($C_{base} \cap C_{novel} = \varnothing$). At the same time, the fact that the novel class is not involved in training makes the testing scenario of the model quite extensive. In the framework of few-shot learning, the learning tasks are organized in the form of N-way K-shots. In each task, the model needs to learn from N novel classes with only K samples. The task consists of a support set S and a query set Q. The support set is the feature benchmark for each category, and the query set consists of labeled supervised samples in training and unlabeled test samples in the final testing phase. The respective definitions are as follows:

$$S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \tag{1}$$

$$Q = \{(x_1, y_1), \ldots, (x_n, y_n)\} \tag{2}$$

where (x_s, y_s) denotes the image and its corresponding label, and m and n are the number of samples contained in the query set and the support set, respectively. During training, the model utilizes these samples for learning and calculates the supervised loss by predicting the labels of the query set.

N-way K-shot means that there are N categories in the support set and only K support samples in each category. Simply put, the parameter K indicates exactly how many samples are in each category in a recognition task with N categories.

2.3. Methods

2.3.1. Baseline Framework for Meta-Learning

As shown in Figure 1, the meta-framework used in this study consists of two parts: the CNN encoder $F(\theta)$ and the metric module. In the pre-training phase, a traditional linear classifier is mounted on the tail of the encoder $F(\theta)$, which is trained using the segmented base class images, and the loss of cross-entropy after classification is computed to supervise the model and to gain some prior knowledge to avoid the model starting from random initialization in the meta-learning phase. The learning objective in the meta-training phase is to learn the ability to transform between tasks. First, the pre-trained model is loaded, and the one-stage linear classifier is removed. Second, after organizing the data into an N-way K-shot task, the loss is computed from the query set, thereby fine-tuning the encoder. Specifically, under the K-shot task, the method extracts the K samples contained in category c in the support set into K high-dimensional feature vectors by the encoder. Subsequently, their average value is computed as the prototype center W_C for representing category c, which can be expressed as follows:

$$W_C = \frac{1}{|S_c|} \sum_{x_s \in S_C} F_{\theta}(x_s) \tag{3}$$

where S_c denotes the number of samples contained in the support set for category c (K). Meanwhile, the samples in the query set Q are processed by the feature extraction function $F(\theta)$, which generates q high-dimensional feature vectors (x_q). Based on this information, the probability that a sample q belongs to category c can be computed with the following expression:

$$p(y = c|x_q) = \frac{\exp(-D(F_\theta(x_q), W_C)}{\sum_{C'} -D(F_\theta(x_q), W_{C'})} \tag{4}$$

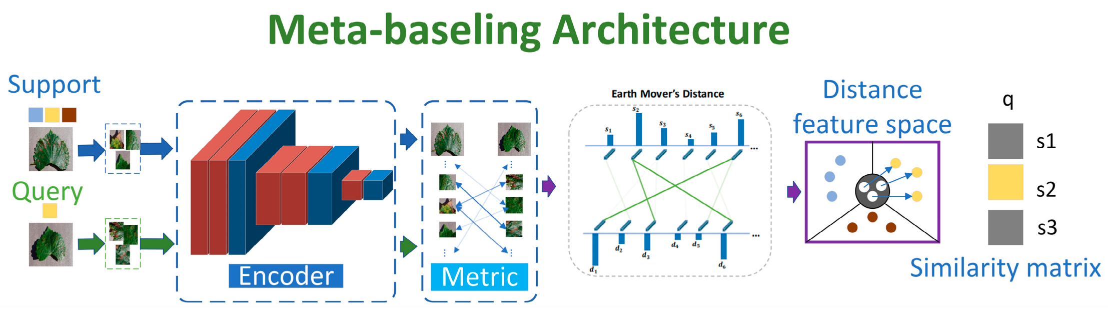


Figure 1. Meta-learning baseline framework diagram.

The metric function selected for this study is EMD (Earth Mover's Distance) [23], which is an image similarity measure. The specific calculation is as follows:

$$A_{ij} = 1 - \frac{{S_i}^T D_j}{||S_i||||D_j||} \tag{5}$$

$$\text{Minimize}: D_{ij} = \sum_{i=1}^{m} \sum_{j=1}^{k} X_{ij} A_{ij} \tag{6}$$

$$Subjectto: \sum_{j}^{k} X_{ij} = S_i \tag{7}$$

$$\sum_{i}^{m} X_{ij} = Q_j \tag{8}$$

where the product of the number of feature maps and the size of the feature maps of the support set images is denoted as k, and the query set is denoted as m. m and k can be considered as the number of locations on the supply and demand sides, respectively. The unit transportation cost A_{ij} between vectors (as shown in Equation (5)) is represented by the cosine distance. Therefore, a linear programming problem with Equations (7) and (8) constraints for EMD is constructed for images between the support and query sets. The optimal matching cost between images (as shown in Equation (6)) can be obtained by controlling the amount of transportation from each supplier i to each demander j. EMD constructs a spatial nearest-neighbor problem under the local feature space of the images by computing the minimum matching cost, as defined in Equation (6), and uses this cost as the distance metric in the prototypical network. Also, for the purpose of distancing different categories from each other, the model is updated using a cross-entropy loss function.

2.3.2. Slimming Network Pruning Algorithm

In this paper, network slimming, an efficient pruning method based on BN layers, is used. During the training of neural networks, the distribution of input data in each layer changes due to parameter updates, which can make training extremely difficult [31]. To solve this problem, Santurkar [32] reduced the internal covariate bias by introducing a BN layer to normalize each small batch of data so that the distribution of the input data in each layer is between 0 and 1, thus reducing this bias. Specifically, the mean (μ_B) and variance

(δ_B^2) of this feature map in the mini-batch are first computed, as well as the normalized value $\widehat{x}_i$ of the input samples:

$$\widehat{x}_i = \frac{x_i - \mu_B}{\sqrt{\delta^2 + \epsilon}} \tag{9}$$

$$\mu_B = \frac{1}{\mathrm{m}} \sum_{i=1}^{m} x_i \tag{10}$$

$$\delta_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{11}$$

where m is the number of samples in a small batch and x_i is the feature map data of the ith sample. In addition, in order to enhance the generalization ability of the network, two learnable parameters, γ (gamma) and β (beta), are introduced in the BN layer to restore the original distribution of the data. The final output y_i of the network is as follows:

$$y_i = \gamma \widehat{x}_i + \beta = \frac{\gamma}{\sqrt{\sigma_B^2 + \epsilon}} \widehat{x}_i + \left(\beta - \frac{\gamma \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \right) \tag{12}$$

where γ is the weight coefficient, β is the bias sparsity, and ε is a very small constant to avoid the numerator being zero. In convolutional neural networks, the BN layer is generally followed by the activation function and the convolutional layer; then, for a new convolutional layer, the output size of the BN layer determines whether the current channel is important or not. In turn, the output size is determined by the value of γ. Therefore, in this study, the size of γ is regarded as an indicator to judge the importance of the channel, and the redundant parameters in the network are reduced by removing the channel with a small value of γ. Figure 2 shows a schematic diagram of the pruning algorithm, which prunes the channels in the network by evaluating them based on the importance evaluation factor γ. This process centers on removing input and output connections that are associated with a particular channel. As shown in Figure 2, channels with smaller scale factors (marked in red) are cropped out. Meanwhile, channels with larger scale factors (indicated in blue) are retained. With this cropping operation, a more compact and efficient neural network structure is finally obtained (in green).

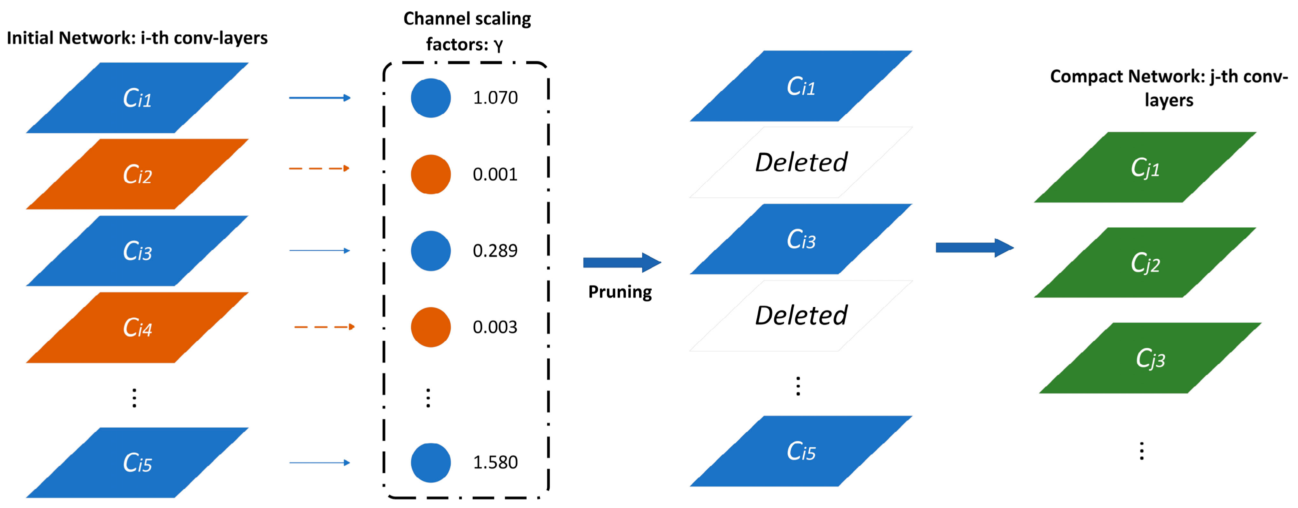

Initial Network: i-th conv-layers
Ci1
Ci2
Ci3
Ci4
Ci5
Channel scaling factors: γ
1.070
0.001
0.289
0.003
1.580
Pruning
Ci1
Deleted
Ci3
Deleted
Ci5
Compact Network: j-th conv-layers
Cj1
Cj2
Cj3

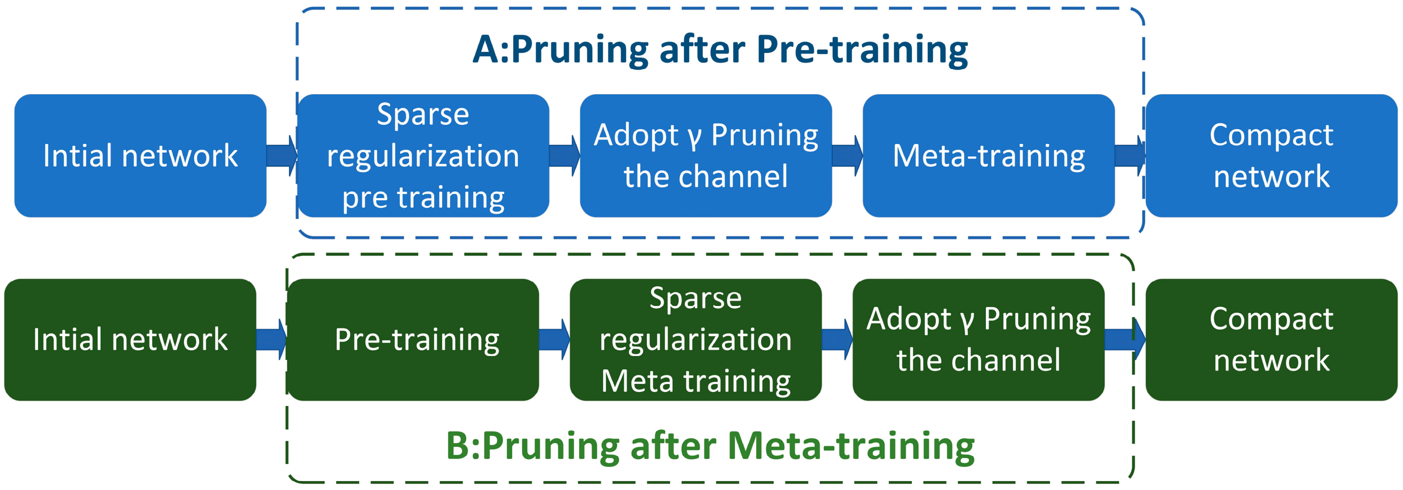
Figure 2. Schematic diagram of the slimming pruning algorithm.

In any network architecture, the scaling factor γ for batch normalization (BN) has a certain weight (the value cannot be exactly 0), which indicates that each channel is crucial for the functioning of the network. In order to deal with this characteristic, this study introduces the sparse training technique, i.e., reducing the value of the γ parameter for a portion of the channels by introducing a regularization term, making the network more concise. This process can be formulated as a loss function:

$$L = \sum l(f(x, W), y) + \lambda \sum g(\gamma) \tag{13}$$

where x denotes the input; y is the true label; W is the trainable parameter in the network; and L1 regularization, as a sparse penalty function, is $g(\gamma)$.

In addition, unlike traditional learning where direct pruning can be performed after sparsifying pre-training, Prune-FSL's framework is divided into two stages: pre-training and meta-learning. Therefore, two pruning strategies were designed in this study. They are used to explore at which stage pruning can be performed to obtain better pruning results. As shown in the blue part of Figure 3, in the strategy of pruning after pre-training, first, the regular pre-training of the network with sparsification is carried out so that a large part of the network γ in the original network is close to 0. Additionally, the channels are pruned. Eventually, the pruned network is meta-trained to obtain a compact network. In the strategy of pruning after meta-training, first, a regular pre-training of the network is performed to obtain a complete model. Subsequently, sparsifying meta-training and pruning the channels are carried out to obtain a sparsified meta-trained model.



A:Pruning after Pre-training
Intial network
Sparse regularization pre training
Adopt γ Pruning the channel
Meta-training
Compact network
Intial network
Pre-training
Sparse regularization Meta training
Adopt γ Pruning the channel
Compact network
B:Pruning after Meta-training

Figure 3. Flowchart of pruning strategy. (**A**) is the process of pruning after pre-training; (**B**) is the process of pruning after meta-training.

2.3.3. Indicators for Model Evaluation

The method proposed in this study is a lightweight small-sample classification model that can be used for small computing devices, such as edge devices. Therefore, the model needs to take into account accuracy while having smaller parameters and smaller computation. In this study, three metrics—accuracy, parameter, and Macs—are chosen to comprehensively evaluate the performance of the model.

Accuracy is a basic metric for evaluating the performance of a classification model. It indicates the number of samples correctly predicted by the model as a proportion of the total number of samples. In a multicategorization problem, the formula for calculating accuracy is as follows:

$$\text{Accuracy} = \frac{\sum_{h=1}^{n} TP_h}{\sum_{h=1}^{n} (TP_h + FN_h + FP_h)} \times 100\% \tag{14}$$

where TP_h (True Positive) is the number of samples correctly predicted by the model as the h-th positive category, FP_h (False Positive) is the number of samples incorrectly predicted by the model as the h-th positive category, and FN_h (False Negative) is the number of samples incorrectly predicted by the model as the h-th negative category. N denotes the total number of categories.

In order to accurately quantify the network parameters, this study introduces the metric "Parameter". Specifically, it refers to the total number of weights and biases to be learned in the model. The number of parameters directly affects the complexity and size of the model and usually also affects the training time and performance of the model. The calculation of the parameters is usually based on the number of layers in the model and the type of operations in each layer. Since the model used has only convolutional and BN layers, the number of parameters for a BN layer plus a convolutional layer can be expressed as follows:

$$\text{Parameter } = \mathrm{C_{out}} \times (\mathrm{K_W} \times \mathrm{K_h} \times \mathrm{C_{in}} + 1) + 2 \times \mathrm{C} \tag{15}$$

where C_{out} denotes the number of output channels and C_{in} expresses the number of input channels. K_h, K_W are the height and width of the convolution kernel.

Macs (Multiply–Accumulate operations) is a metric used in deep learning to measure the computational complexity of a model. A Multiply–Accumulate operation means that a multiplication operation is immediately followed by an addition operation. In deep learning, models such as convolutional neural networks (CNNs) and Recurrent Neural Networks (RNNs) rely heavily on a large number of matrix multiplication and accumulation operations to process and analyze complex data patterns, and thus Macs are used to estimate the computational resources required by the model for inference or training. Their computation is usually based on the number of layers of the model and the type of operations in each layer. Since in Prune-FSL only the convolutional layers in the backbone network are used, the Macs metric is calculated as follows:

$$\text{Macs } = H \times W \times C_{In} \times C_{out} \times K_W \times K_h \tag{16}$$

where H represents the height of the feature map and W represents the convolution kernel width.

3. Results

3.1. Data Setting

In order to fulfill the requirement of mutual exclusivity of categories in the training and test sets of FSL, this study slices Plantvillage into three parts according to the distribution of categories in Figure 4: the training set, the validation set, and the test set. The training set contains 19 categories that are directly involved in the gradient update of the model. The validation set contains ten categories that are used to monitor the model's performance during training and keep the best weights. The remaining nine categories serve as the test set to examine the final performance criteria of the model.

3.2. Effect of Pruning Coefficients on Model Performance

Prune-FSL employs a γ-value-based channel pruning strategy, i.e., all the γ-values in the network are first sorted, and then the pruning threshold is determined based on the preset pruning ratio coefficients, and those channels with γ-values lower than this threshold are pruned [24,33]. In this experiment, four pruning scale coefficients, 0.2, 0.4, 0.6, and 0.8 [34], are carefully designed to explore the effects of different compression levels on the model performance. After the pruning process, four lightweight network models were obtained: Resnet12—20%, Resnet12—40%, Resnet12—60%, and Resnet12—80%. The performance of these models on the 5-way 1-shot and 5-way 5-shot tasks is detailed in Table 2. The experimental results show that with a pruning ratio of 20%, i.e., the Resnet12—20% model, the number of parameters is reduced to a minimum of 0.49 M, which is as much as a 25-fold reduction compared to the original network; at the same time, the amount of computation is also reduced significantly from 3.52 G to 0.14 G. Especially noteworthy is that the performance loss of the model is only 1.57% under 5-way 5-shot conditions, which fully demonstrates the great potential of the pruning-based lightweight FSL method in achieving efficient small-sample recognition. In addition, this study also provides an in-depth analysis of the trend of the model performance with respect to the parameters. The relevant results are displayed in Figure 5. From the figure, it can be clearly observed that the performance loss is almost negligible when the model pruning coefficient is above 0.6, regardless of the 5-way 1-shot (blue curve) or 5-way 5-shot (red curve) conditions, even though the number of model parameters is reduced by nearly three times. This provides a practical solution for deploying high-performance FSL models on small computing devices with relatively generous computational resources. However, when the pruning ratio is further increased, the model performance shows a more obvious drop, but even in this case, its performance under 5-way 5-shot conditions remains at an applicable level, showing that the method is still robust under extreme compression conditions.

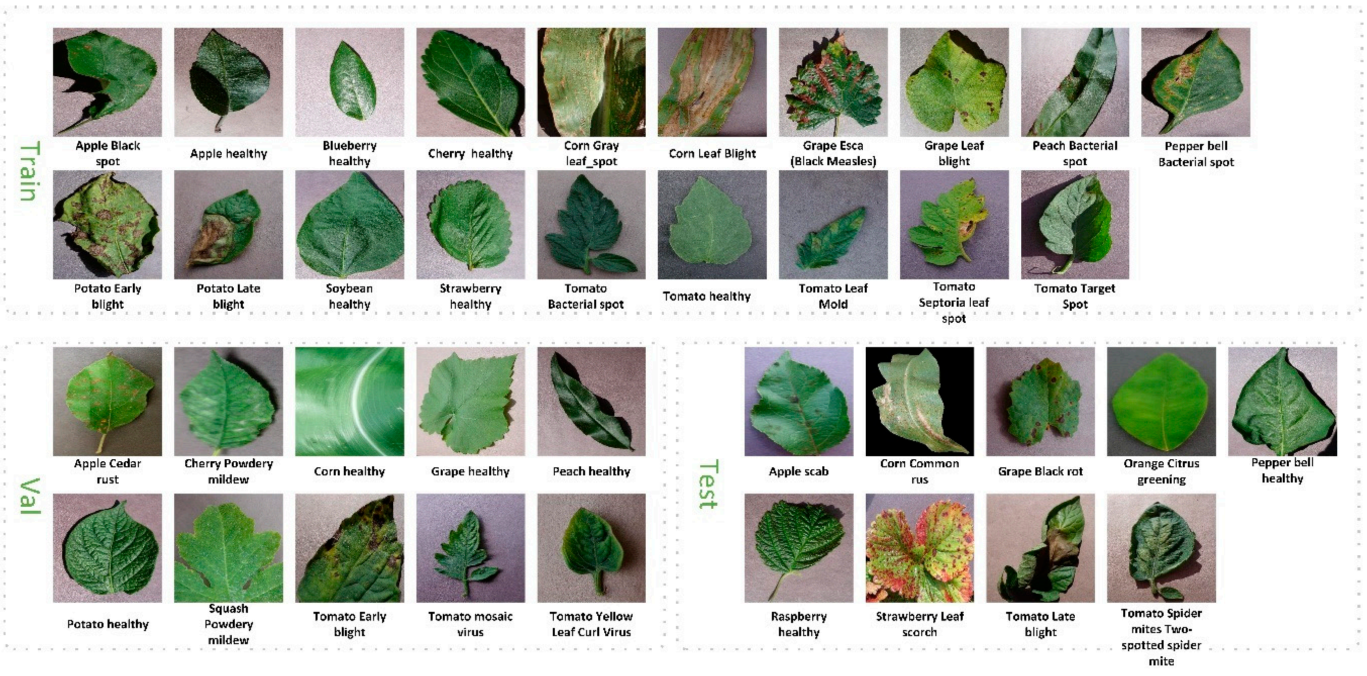


Figure 4. Dataset segmentation chart.

Table 2. Effect of pruning coefficient on performance.

Network	Training Stage	Paramars (M)	Macs (G)	5-way 1-shot		5-way 5-shot	
				Acc (%)	95% MSE	Acc (%)	95% MSE
Resnet12—80%	A	0.49	0.14	77.97	1.46	90.70	1.02
Resnet12—60%	A	1.99	0.56	80.87	1.32	91.37	0.98
Resnet12—40%	A	4.47	1.27	82.07	1.30	92.30	0.90
Resnet12—20%	A	7.95	2.25	82.10	1.31	92.10	0.93
Resnet12—0%	A	12.42	3.52	82.47	1.30	92.27	0.91

3.3. *Effect of Pruning Strategy on Performance*

The baseline approach is composed of pre-training and meta-training [35]. To investigate the effectiveness of different pruning strategies in lensless learning, two pruning strategies were designed in this study [36]: (A) pruning after meta-training and (B) pruning after sparse pre-training (Figure 6). By controlling the change in pruning strategy, a comparison test was conducted for the four post-pruning models. The specific results are shown in Table 3. The experimental results show that the accuracy of both training strategies A and B increases as the percentage of pruning is reduced from 80% to 20%. This indicates that the performance of the model benefits from a higher pruning percentage. At the same time, the models in strategy B generally do not perform as well as in strategy A, and this gap becomes more pronounced the smaller the parameters are. In conclusion, training strategy A generally outperforms strategy B, especially when a larger pruning ratio is used.

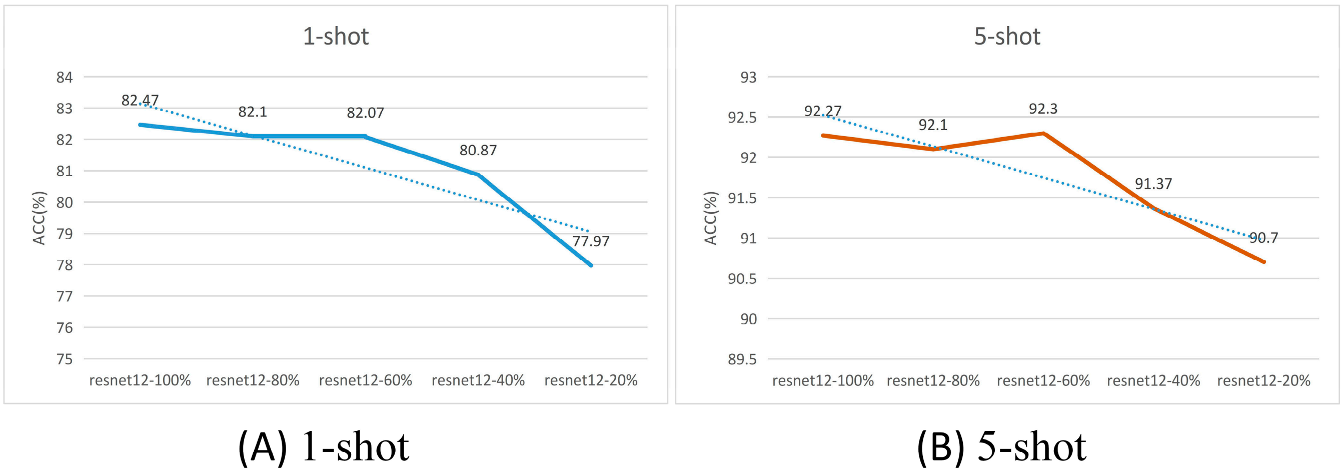


Figure 5. Trend of model performance with parameters. (**A**) is the trend in the 1-shot; (**B**) is the trend in the 5-shot. Note: The dashed line in the figure shows the general trend in accuracy; the orange line represents the change in accuracy in the 5-shot condition, and the blue line represents the change in accuracy in the 1-shot condition.

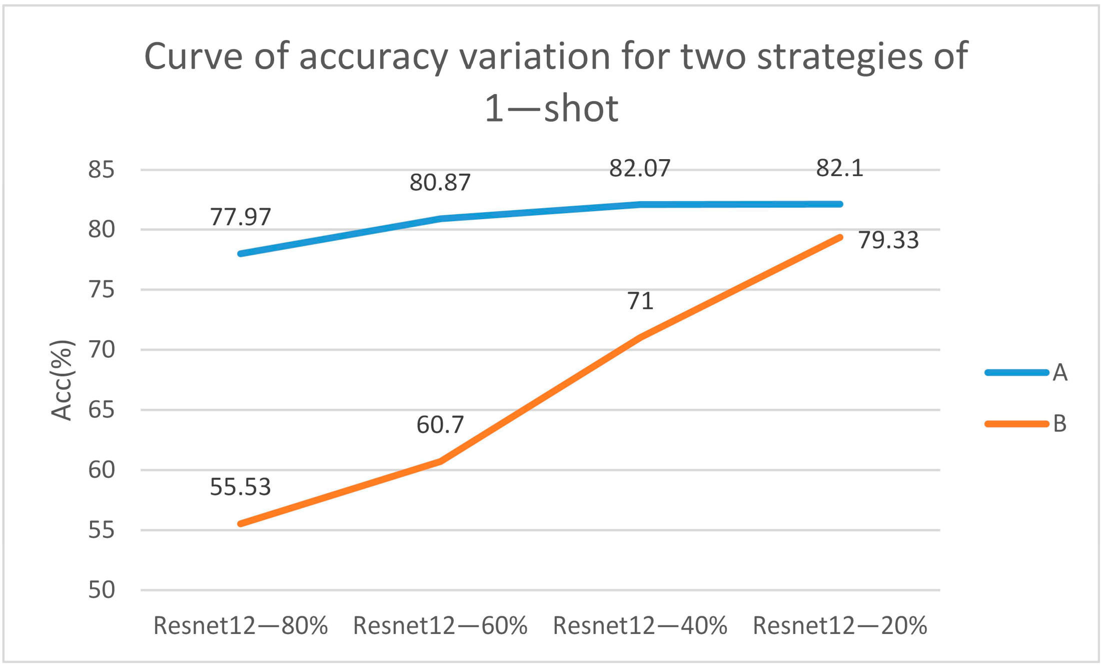


Figure 6. Trend of two strategies with parameters. (A) reflects the curve of accuracy as a function of pruning ratio under the strategy of pruning after pre-training; (B) reflects the curve of accuracy as a function of pruning ratio under the process strategy of pruning after meta-training.

Table 3. Impact of pruning strategy on performance.

Network	**Training Stage**	**5-way 1-shot**		**5-way 5-shot**	
		Acc (%)	**95% MSE**	**Acc (%)**	**95% MSE**
Resnet12—80%	A	77.97	1.46	90.70	1.02
Resnet12—60%	A	80.87	1.32	91.37	0.98

Table 3. *Cont.*

Network	Training Stage	5-way 1-shot		5-way 5-shot	
		Acc (%)	**95% MSE**	**Acc (%)**	**95% MSE**
Resnet12—40%	A	82.07	1.30	92.30	0.90
Resnet12—20%	A	82.10	1.31	92.10	0.93
Resnet12—80%	B	55.53	1.72	65.77	1.65
Resnet12—60%	B	60.70	1.69	69.13	1.55
Resnet12—40%	B	71.00	1.53	81.73	1.37
Resnet12—20%	B	79.33	1.42	88.60	1.12

3.4. Visualization of Loss Curves

The model training process is divided into two phases (pre-training and meta-training), and Resnet12—20% is chosen as the main research object of this section. To visualize the training process more clearly [37], the loss function and accuracy are visualized in this study. The specific loss function trends are revealed in Figure 7. In Figure 7A,B, the loss function change curves of the model during the pre-training and meta-training phases [38] are shown, respectively. It is worth noting that the training loss exhibited by the model at the beginning of meta-training is still significant despite the completion of the pre-training phase. This phenomenon reflects that the main role of pre-training is to provide a priori knowledge for the model, and the feature space differences that have not been tuned by the N-way K-shot meta-learning are not yet fully revealed. However, as the meta-training proceeds further, the differentiation of different samples in the feature space gradually increases, and meta-training loss shows a steadily decreasing trend, eventually converging to a level of about 0.4. This process is usually completed within 40 rounds of iterations, thus confirming that the model possesses good generalization ability. In addition, Figure 7C,D depict the accuracy change curves of the validation set in the pre-training and meta-training phases, respectively. From the figures, it can be clearly observed that the highest accuracy of the validation set reaches 86.00% and 82.50%, respectively. This result demonstrates a rapid improvement in the performance of the model on the validation set.

3.5. K-Fold Cross-Validation

In order to more rigorously verify the robustness and generalization ability of the model, this study adopts the K-Fold cross-validation method to systematically evaluate the pruned 80% model. In this process, the training dataset was uniformly divided into five subsets (labeled as 1, 2, 3, 4, and 5) [39], each of which was regarded as an independent fold. In this study, each fold was trained separately, and five sets of accuracy data were collected. In order to obtain more reliable evaluation results, this study averaged these five sets of accuracy data, and the average value obtained was used as the accuracy rate of K-Fold cross-validation. The specific experimental results are detailed in Table 4. After five cross-validations, the model achieved 77.17% and 88.86% accuracy on the 5-way 1-shot and 5-way 5-shot tasks, respectively. The 95% mean square error [40] of the cross-validation results was also calculated in this study, which was 1.42% and 1.11%, respectively. This result indicates that the model is able to maintain relatively stable performance under different data partitioning and training conditions, thus confirming its strong robustness and generalization ability. This is crucial for the reliability and stability of the model in practical applications and provides strong support for further promotion and application of the model.

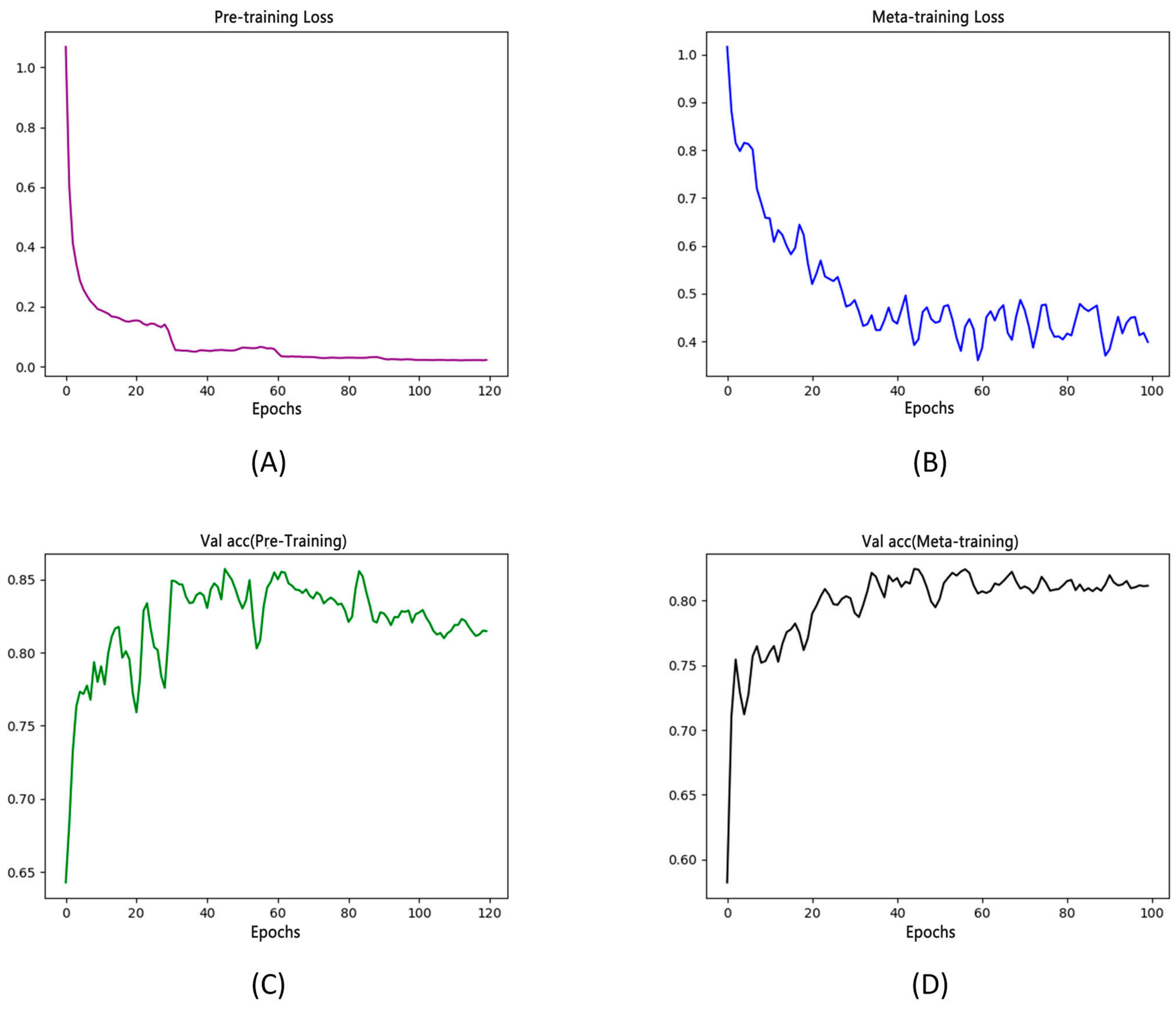


Figure 7. Visualization of loss curve vs acc curve. (**A**) is the loss curve of the pre-training; (**B**) is the loss curve of the meta-training; (**C**) is the acc curve of the pre-training; (**D**) is the acc curve of the meta-training.

Table 4. K-Fold cross-validation table.

Network	**Fold**	**5-way 1-shot**		**5-way 5-shot**	
		Acc (%)	**95% MSE**	**Acc (%)**	**95% MSE**
Resnet12—20%	1st fold	73.00	1.54	89.40	1.07
Resnet12—20%	2nd fold	76.70	1.32	90.37	1.07
Resnet12—20%	3th fold	79.77	1.41	87.47	1.19
Resnet12—20%	4th fold	77.77	1.42	87.43	1.15
Resnet12—20%	5th fold	78.60	1.39	89.63	1.06
Resnet12—20%	AVG	77.17	1.42	88.86	1.11

3.6. Comparison with Lightweight Networks

In order to deeply explore the utility of pruning-based approaches in the field of few-shot learning (FSL) and to compare them with approaches that only lighten the backbone network by replacement, this paper conducts a series of studies under the same experimental conditions. Specifically, this paper replaces different backbone networks in the baseline model and conducts a total of seven sets of experiments under two task settings: 5-way 1-shot and 5-way 5-shot. The experimental results are detailed in Table 5. Analyzing the data, it can be seen that EfficientNet performs well in terms of computational efficiency, with a computational volume of only 0.03 G, and also achieves 73.60% and 85.70% in terms of accuracy. Shufflenetv2, on the other hand, achieves considerable performance while maintaining a low number of parameters. Notably, Densenet40 stands out with its 0.61 M parameter count in the 1-shot task, achieving the best performance except for the baseline model, and even surpassing the baseline model in the 5-shot task. However, its 0.82 G computation is still high compared to the pruning model. On the other hand, Mobilenetv2 and Mobilenetv3 perform relatively poorly in this experiment, not only in terms of poor performance but also in terms of a larger number of parameters. In contrast, Resnet12—80% shows excellent comprehensive performance, which can still significantly outperform other lightweight network models while significantly reducing the network volume, and its performance is only second to that of Densenet40. To summarize, the experimental results in this paper fully demonstrate the superiority of the pruning-based lightweight FSL method in terms of performance. Through a well-designed pruning strategy, the model in this paper can effectively reduce the computational and parametric quantities while guaranteeing the model performance, which provides new ideas and directions for the development of the FSL field.

Table 5. Comparison test with lightweight networks.

Network	**Paramars (M)**	**Macs (G)**	**5-way 1-shot**		**5-way 1-shot**	
			Acc(%)	**95% MSE**	**Acc(%)**	**95% MSE**
Shufflenetv2	1.99	0.03	73.60	1.55	87.70	1.18
Mobilenetv2	3.50	0.05	72.67	1.45	85.90	1.27
Mobilenetv3	5.76	0.05	67.60	1.62	82.30	1.37
EfficientNet	2.82	0.03	73.60	1.50	85.70	1.25
Densenet40	0.61	0.82	79.63	1.41	94.40	0.80
Resnet12	12.42	3.52	82.47	1.30	92.27	0.91
Resnet12—80%	0.49	0.14	77.97	1.46	90.70	1.02

3.7. Comparison with Related Work

In order to fully verify the validity of the methodology of this study, we refer to the typical representative works in the field of "few-shot learning" from 2017 to 2023, using the timeline as a variable. Prototypical networks [41], matching networks [42], DeepEMD [23], FEAT [43], and FewTURE [44] were carefully selected to conduct systematic comparative experiments in this study. All experiments were conducted under strictly uniform data conditions to ensure the fairness and comparability of the results. The experimental results are shown in Table 6 and Figure 8. After 80% network pruning, the method still achieves 77.97% accuracy under 1-shot conditions, a result that is comparable to the prototypical network model in 2018. More notably, the method further improved the accuracy to 90.7% under 5-shot conditions, a remarkable result that not only rivals the performance of traditional FSL models but also signifies that it has reached the practical level in relevant applications such as crop disease identification. In summary, the algorithms in this study have initially achieved the goal of lightweighting in the field of few-shot learning, especially in crop disease recognition. This important progress not only provides new ideas for the optimization of related algorithms but also lays a solid foundation for future promotion and deployment in practical applications.

Table 6. Comparison of related work.

Method	**5-way 1-shot**		**5-way 5-shot**	
	Acc (%)	**95% MSE**	**Acc (%)**	**95% MSE**
Matching net (2017)	81.52	0.01	91.14	0.01
Prototypical net (2018)	78.21	1.56	89.47	1.06
FEAT (2021)	81.38	1.35	91.29	0.96
DeepEMD (2023)	82.47	1.30	92.27	0.91
FewTURE (2023)	84.20	1.21	95.53	0.74
Resnet12—20%	77.97	1.46	90.70	1.02

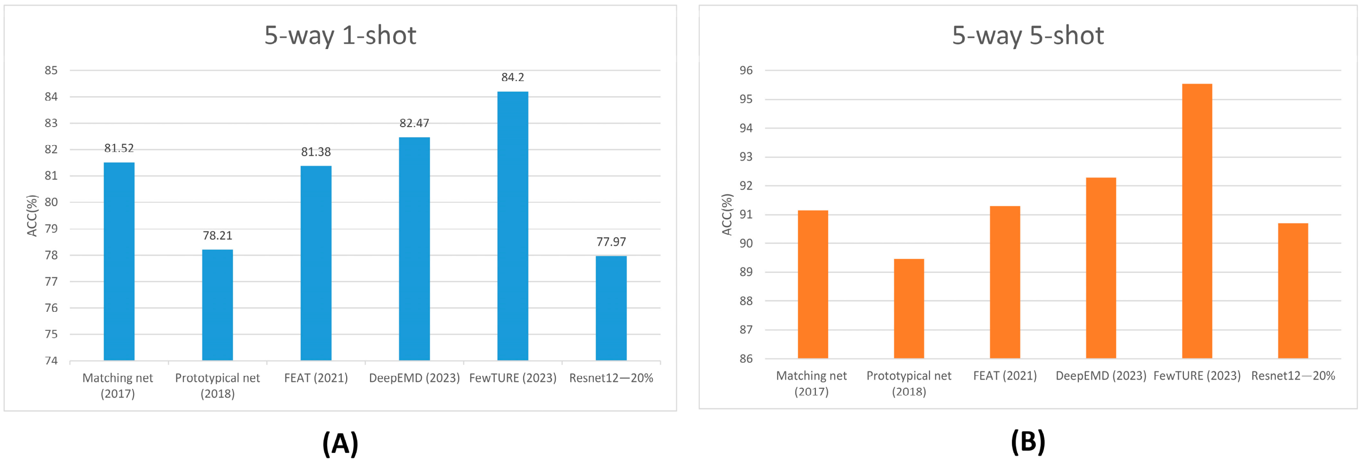


Figure 8. Comparative bar charts of related work. (**A**) is a histogram of the accuracy of the different models under the 5-way 1-shot task; (**B**) is a histogram of the accuracy of the different models under the 5-way 5-shot task.

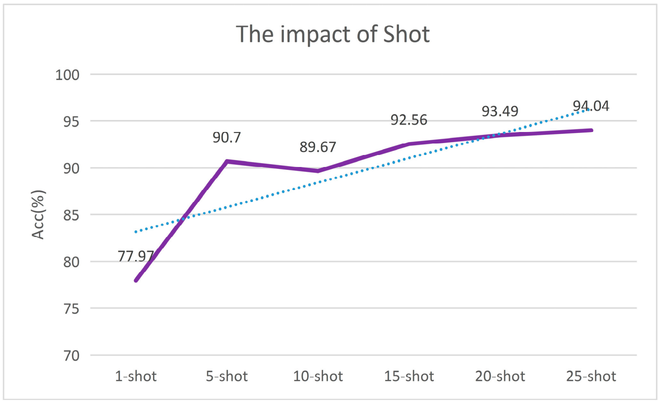
4. Discussion

4.1. Limitations of the Model and Future Work

First, a single crop may suffer from multiple diseases. Although each image of the Plantvillage dataset used in this study depicts only one disease, in reality, once a crop is infected with one disease, its immune system is weakened, making it more susceptible to other diseases [45,46]. This is especially common in field environments where the same plant may suffer from multiple diseases at the same time. From a classification point of view, the feature information of the image may become blurred [47]. To solve this problem, we choose to utilize multimodal information [48], such as text (e.g., "image of potato with early blight") or audio, in order to correctly instruct the image encoder to learn this part of the knowledge and to prune it after learning is complete. Second, the importance of the cross-domain problem in small-sample learning cannot be ignored. It is a common scenario that when unpredictable external factors are present during the testing phase, such as disease category, disease severity, weather, and lighting, the performance of the model may experience an unexpected degradation [49,50]. However, the dataset used in this study did not contain this information. Therefore, future work will focus on constructing a more complex dataset in order to test the model's ability to generalize in complex disturbance environments and make further improvements [51].

4.2. Effect of Way and Shot

The parameters N-way and K-shot significantly influence model performance, with N-way denoting the number of classes and K-shot specifying the size of the support set per class. To explore the impact of these parameters on model performance [30,52], we systematically varied their values and conducted a comprehensive statistical analysis of the accuracy of the Resnet12—20% model. As illustrated in Figure 9, the model's accuracy exhibits a marked improvement as the K-shot value increases from 1 to 25, rising from 77.97% to 94.04%. Notably, the increase in accuracy stabilizes once the K-shot value surpasses 10, indicating an effective enhancement of model performance through the augmentation of prior information.



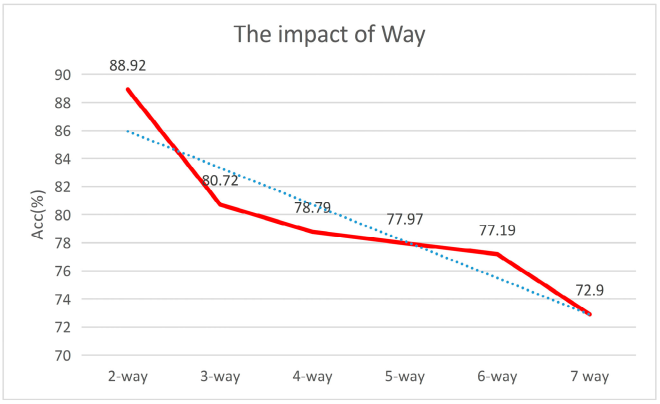


Figure 9. Way and shot trend chart.(**A**) is the impact of shot;(**B**) is the impact of way. Note: The dashed line in the figure shows the general trend in accuracy; The purple line indicates the change in model accuracy with shot and the red line indicates the change in model accuracy with way.

Conversely, an increase in N-way introduces greater diversity and, consequently, higher information entropy, which negatively affects model accuracy. Therefore, we advocate for a strategy that balances model lightness with an increase in the support set size, particularly in scenarios characterized by a broad range of categories. This approach ensures the maintenance of or improvement in model accuracy while managing computational complexity [53].

5. Conclusions

To address edge computing device limitations and the generalization issues of few-shot learning for crop disease recognition, we propose Prune-FSL. This network reduces parameters and enhances recognition accuracy. Using the Plantvillage dataset, we evaluated the model through ablation studies, cross-validation, and comparative experiments, yielding the following conclusions:

(1) Feasibility analysis of FSL pruning: The experiments show that incorporating the pruning method into the feature extraction network of the meta-learner can reduce the computation of the model by up to 25 times, and the performance of the model is still maintained at a high level. This shows that the method can still maintain good performance under extreme compression conditions, and the performance of the model varies with the compression ratio—the higher the compression rate, the lower the performance.

(2) In the comparison of the two pruning strategies, this study found that when the pruning ratios were kept the same, strategies that were pruned after sparse pre-training consistently showed better performance. In addition, the difference in performance between the two strategies became more significant as the pruning ratio increased. In addition, this study conducted five-fold cross-validation experiments with loss function visualization for networks with pruning ratios up to 80%, which not only verified the robustness of the model but also further enhanced its persuasiveness.

(3) In the final comparison experiment, this study compares this paper's model with several lightweight networks, and the results show that the Resnet12—80% model, which is significantly better than other lightweight models, still exhibits excellent overall performance while reducing the network size. In addition, this study also compares our model

with several representative few-shot learning (FSL) methods proposed in recent years. With highly compressed models, Prune-FSL achieves performance comparable to that of Resnet12-based matching networks and prototypical networks.

In conclusion, the method proposed in this study provides an effective solution for lensless crop disease recognition under computational resource constraints and provides techniques and references for subsequent lensless crop disease recognition on field plant-protection robots.

Author Contributions: W.Y. (Wenbo Yan): Conceptualization, Formal analysis, Methodology, Visualization, and Writing—original draft preparation; Q.F.: Writing—review and editing, Funding acquisition, Validation, Resources, and Software; S.Y.: Data curation, Writing—review and editing, Funding acquisition, Validation, Resources, Software, and Project administration; J.Z.: Validation and Writing—review and editing; W.Y. (Wanxia Yang): Investigation and Software. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (Grant No. 32201663, 32160421, 31971792), funding from the Industrialization Support Project from Education Department of Gansu Province (Grant No. 2021CYZC-57).

Data Availability Statement: The Plantvillage dataset is publicly available online at https://github.com/spMohanty/PlantVillage-Dataset (accessed on 28 September 2018).

Acknowledgments: I would like to thank the technicians who assisted me with the data collection and analysis for their invaluable support.

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

References

1. Strange, R.N.; Scott, P.R. Plant disease: A threat to global food security. *Annu. Rev. Phytopathol.* **2005**, *43*, 83–116. [CrossRef] [PubMed]
2. Oerke, E.C.; Dehne, H.W. Safeguarding production—Losses in major crops and the role of crop protection. *Crop Prot.* **2004**, *23*, 275–285. [CrossRef]
3. He, D.-c.; ZHAN, J.-s.; Xie, L.-h. Problems, challenges and future of plant disease management: From an ecological point of view. *J. Integr. Agric.* **2016**, *15*, 705–715. [CrossRef]
4. Khakimov, A.; Salakhutdinov, I.; Omolikov, A.; Utaganov, S. Traditional and current-prospective methods of agricultural plant diseases detection: A review. *IOP Conf. Ser. Earth Environ. Sci.* **2022**, *951*, 012002. [CrossRef]
5. Ma, J.; Du, K.; Zheng, F.; Zhang, L.; Gong, Z.; Sun, Z. A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. *Comput. Electron. Agric.* **2018**, *154*, 18–24. [CrossRef]
6. Zhang, X.; Qiao, Y.; Meng, F.; Fan, C.; Zhang, M. Identification of Maize Leaf Diseases Using Improved Deep Convolutional Neural Networks. *IEEE Access* **2018**, *6*, 30370–30377. [CrossRef]
7. Fuentes, A.; Yoon, S.; Kim, S.C.; Park, D.S. A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition. *Sensors* **2017**, *17*, 2022. [CrossRef]
8. Dhiman, P.; Kaur, A.; Balasaraswathi, V.R.; Gulzar, Y.; Alwan, A.A.; Hamid, Y. Image Acquisition, Preprocessing and Classification of Citrus Fruit Diseases: A Systematic Literature Review. *Sustainability* **2023**, *15*, 9643. [CrossRef]
9. Long, M.S.; Ouyang, C.J.; Liu, H.; Fu, Q. Image recognition of Camellia oleifera diseases based on convolutional neural network & transfer learning. *Trans. CSAE* **2018**, *34*, 194–201. [CrossRef]
10. Chen, J.; Ran, X. Deep learning with edge computing: A review. *Proc. IEEE* **2019**, *107*, 1655–1674. [CrossRef]
11. Su, D.; Deng, Y.Z. Research progress and problems in crop disease image recognition. *J. Tianjin Agric. Univ.* **2023**, *30*, 75–79. [CrossRef]
12. Véstias, M.P. Deep learning on edge: Challenges and trends. In *Smart Systems Design, Applications, and Challenges*; IGI Global: Hershey, PA, USA, 2020; pp. 23–42.
13. Kamath, V.; Renuka, A. Deep learning based object detection for resource constrained devices: Systematic review, future trends and challenges ahead. *Neurocomputing* **2023**, *531*, 34–60. [CrossRef]
14. Hu, G.S.; Wu, H.Y.; Zhang, Y.; Wan, M.Z. A low shot learning method for tea leaf's disease identification. *Comput. Electron. Agric.* **2019**, *163*, 104852. [CrossRef]
15. Chen, Y.P.; Pan, J.C.; Wu, Q.F. Apple leaf disease identification via improved CycleGAN and convolutional neural network. *Soft Comput.* **2023**, *27*, 9773–9786. [CrossRef]
16. Cap, Q.H.; Uga, H.; Kagiwada, S.; Iyatomi, H. LeafGAN: An Effective Data Augmentation Method for Practical Plant Disease Diagnosis. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 1258–1267. [CrossRef]
17. Pan, S.J.; Yang, Q.A. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]

18. Li, Z.J.; Xu, J.; Zeng, L.; Tie, J.; Yu, S. Small sample recognition method of tea disease based on improved DenseNet. *Trans. CSAE* **2022**, *38*, 182–190. [CrossRef]
19. Yang, M.X.; Zhang, Y.G.; Liu, T. Corn disease recognition based on the Convolutional Neural Network with a small sampling size. *Chin. J. Eco-Agric.* **2020**, *28*, 1924–1931. [CrossRef]
20. Xiao, W.; Quan, F. Research on plant disease identification based on few-shot learning. *J. Chin. Agric. Mech.* **2021**, *42*, 138–143.
21. Lin, H.; Tse, R.; Tang, S.K.; Qiang, Z.P.; Pau, G. Few-shot learning approach with multi-scale feature fusion and attention for plant disease recognition. *Front. Plant Sci.* **2022**, *13*, 907916. [CrossRef]
22. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
23. Zhang, C.; Cai, Y. DeepEMD: Differentiable Earth Mover's Distance for Few-Shot Learning. *arXiv* **2023**, arXiv:2003.06777v5. [CrossRef]
24. Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; Zhang, C. Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2736–2744.
25. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
26. 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.
27. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
28. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
29. Hughes, D.; Salathé, M. An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv* **2015**, arXiv:1511.08060.
30. Lake, B.M.; Salakhutdinov, R.; Tenenbaum, J.B. Human-level concept learning through probabilistic program induction. *Science* **2015**, *350*, 1332–1338. [CrossRef]
31. Han, S.; Mao, H.; Dally, W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv* **2015**, arXiv:1510.00149.
32. Santurkar, S.; Tsipras, D.; Ilyas, A.; Madry, A. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems*; NeurIPS: San Diego, CA, USA, 2018; Volume 31.
33. Yang, C.; Liu, H. Channel pruning based on convolutional neural network sensitivity. *Neurocomputing* **2022**, *507*, 97–106. [CrossRef]
34. Molchanov, P.; Mallya, A.; Tyree, S.; Frosio, I.; Kautz, J. Importance estimation for neural network pruning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 11264–11272.
35. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1126–1135.
36. Guo, Y.; Yao, A.; Chen, Y. Dynamic network surgery for efficient dnns. In *Advances in Neural Information Processing Systems*; NeurIPS: San Diego, CA, USA, 2016; Volume 29.
37. Diederik, P.K.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
38. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*; NeurIPS: San Diego, CA, USA, 2014; Volume 27.
39. Bengio, Y.; Grandvalet, Y. No unbiased estimator of the variance of k-fold cross-validation. In *Advances in Neural Information Processing Systems*; NeurIPS: San Diego, CA, USA, 2003; Volume 16.
40. Gareth, J.; Daniela, W.; Trevor, H.; Robert, T. *An Introduction to Statistical Learning: With Applications in R.*; Spinger: Berlin/Heidelberg, Germany, 2013.
41. Snell, J.; Swersky, K.; Zemel, R.S. Prototypical Networks for Few-shot Learning. *arXiv* **2017**, arXiv:1703.05175.
42. Vinyals, O.; Blundell, C. Matching Networks for One Shot Learning. *arXiv* **2017**, arXiv:1606.04080v2.
43. Ye, H.-J.; Hu, H.; Zhan, D.-C.; Sha, F. Few-shot learning via embedding adaptation with set-to-set functions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 8808–8817.
44. Hiller, M.; Ma, R.; Harandi, M.; Drummond, T. Rethinking generalization in few-shot classification. In *Advances in Neural Information Processing Systems*; NeurIPS: San Diego, CA, USA, 2022; Volume 35, pp. 3582–3595.
45. Barbedo, J.G.A. A review on the main challenges in automatic plant disease identification based on visible range images. *Biosyst. Eng.* **2016**, *144*, 52–60. [CrossRef]
46. Alves-Júnior, M.; Alfenas-Zerbini, P.; Andrade, E.C.; Esposito, D.A.; Silva, F.N.; da Cruz, A.C.F.; Ventrella, M.C.; Otoni, W.C.; Zerbini, F.M. Synergism and negative interference during co-infection of tomato and Nicotiana benthamiana with two bipartite begomoviruses. *Virology* **2009**, *387*, 257–266. [CrossRef] [PubMed]
47. Ferentinos, K.P. Deep learning models for plant disease detection and diagnosis. *Comput. Electron. Agric.* **2018**, *145*, 311–318. [CrossRef]

48. Zhou, J.; Li, J.; Wang, C.; Wu, H.; Zhao, C.; Teng, G. Crop disease identification and interpretation method based on multimodal deep learning. *Comput. Electron. Agric.* **2021**, *189*, 106408. [CrossRef]
49. Li, Y.; Chao, X. Semi-supervised few-shot learning approach for plant diseases recognition. *Plant Methods* **2021**, *17*, 68. [CrossRef]
50. Nuthalapati, S.V.; Tunga, A. Multi-domain few-shot learning and dataset for agricultural applications. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 1399–1408.
51. Yan, K.; Guo, X.; Ji, Z.; Zhou, X. Deep transfer learning for cross-species plant disease diagnosis adapting mixed subdomains. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2021**, *20*, 2555–2564. [CrossRef]
52. Wang, J.; Jiang, J. Learning across tasks for zero-shot domain adaptation from a single source domain. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 6264–6279. [CrossRef]
53. Ravi, S.; Larochelle, H. Optimization as a model for few-shot learning. In Proceedings of the International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016.