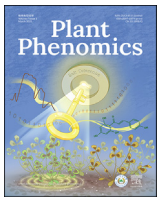
Plant Phenomics 7 (2025) 100024

Contents lists available at ScienceDirect

Plant Phenomics

journal homepage: www.sciencedirect.com/journal/plant-phenomics



Research Article

PlantCaFo: An efficient few-shot plant disease recognition method based on foundation models

Xue Jiang [a], Jiashi Wang [a], Kai Xie [a], Chenxi Cui [a], Aobo Du [a], Xianglong Shi [a], Wanneng Yang [b], Ruifang Zhai [a,c,d,*]

[a] College of Informatics, Huazhong Agricultural University, Wuhan 430070, PR China
[b] National Key Laboratory of Crop Genetic Improvement, National Center of Plant Gene Research, Huazhong Agricultural University, Wuhan 430070, PR China
[c] Engineering Research Center of Intelligent Technology for Agriculture, Ministry of Education, Huazhong Agricultural University, Wuhan 430070, PR China
[d] Hubei Key Laboratory of Agricultural Bioinformatics, College of Informatics, Huazhong Agricultural University, Wuhan 430070, PR China

ARTICLE INFO

Keywords:
Plant disease recognition
Few-shot learning
Foundation models
Efficient funning

ABSTRACT

Although plant disease recognition is highly important in agricultural production, traditional methods face challenges due to the high costs associated with data collection and the scarcity of samples. Few-shot plant disease identification tasks, which are based on transfer learning, can learn feature representations from a small amount of data; however, most of these methods require pretraining within the relevant domain. Recently, foundation models have demonstrated excellent performance in zero-shot and few-shot learning scenarios. In this study, we explore the potential of foundation models in plant disease recognition by proposing an efficient few-shot plant disease recognition model (PlantCaFo) based on foundation models. This model operates on an end-to-end network structure, integrating prior knowledge from multiple pretraining models. Specifically, we design a lightweight dilated contextual adapter (DCon-Adapter) to learn new knowledge from training data and use a weight decomposition matrix (WDM) to update the text weights. We test the proposed model on a public dataset, PlantVillage, and show that the model achieves an accuracy of 93.53 % in a "38-way 16-shot" setting. In addition, we conduct experiments on images collected from natural environments (Cassava dataset), achieving an accuracy improvement of 6.80 % over the baseline. To validate the model's generalization performance, we prepare an out-of-distribution dataset with 21 categories, and our model notably increases the accuracy of this dataset. Extensive experiments demonstrate that our model exhibits superior performance over other models in few-shot plant disease identification.

1. Introduction

The automatic recognition of plant diseases is crucial for ensuring food security and improving yield [1–3]. Considerable progress has been achieved in this area because of advances in large neural architectural design and large-scale labeled data [4–7]. However, this reliance presents significant challenges in agriculture. One challenge is that the collection and annotation of agricultural data are often expensive and time-intensive. Furthermore, the rarity of certain plant diseases makes gathering a large number of examples impractical. Therefore, it is necessary to develop fast and accurate plant disease recognition methods to alleviate the bottlenecks caused by this dependency.

To overcome this bottleneck, an effective solution is to train models using only a small number of labeled samples, a technique called few-shot learning [8–10]. In few-shot learning, datasets are designed to include only a small number of labeled examples for each class, often in the form of a support set and a query set. The support set contains a few labeled examples that the model uses to learn, whereas the query set is used to evaluate the model's ability to generalize. The key evaluation framework in few-shot learning is the N-way K-shot setup. In this framework, N-way refers to the number of distinct classes (e.g., N different plant diseases), and K-shot indicates the number of labeled samples available for each class (e.g., K labeled samples for each disease). For example, in a 5-way 1-shot setting, the model is tasked with classifying among five classes, with only one labeled example per class available for training. Significant progress has been made in this area, primarily through three approaches: data augmentation, meta-learning and transfer learning. Data augmentation enriches the training data

* Corresponding author. College of Informatics, Huazhong Agricultural University, Wuhan 430070, PR China.
E-mail address: rfzhai@mail.hzau.edu.cn (R. Zhai).

https://doi.org/10.1016/j.plaphe.2025.100024
Received 19 July 2024; Received in revised form 15 December 2024; Accepted 10 January 2025
Available online 28 February 2025

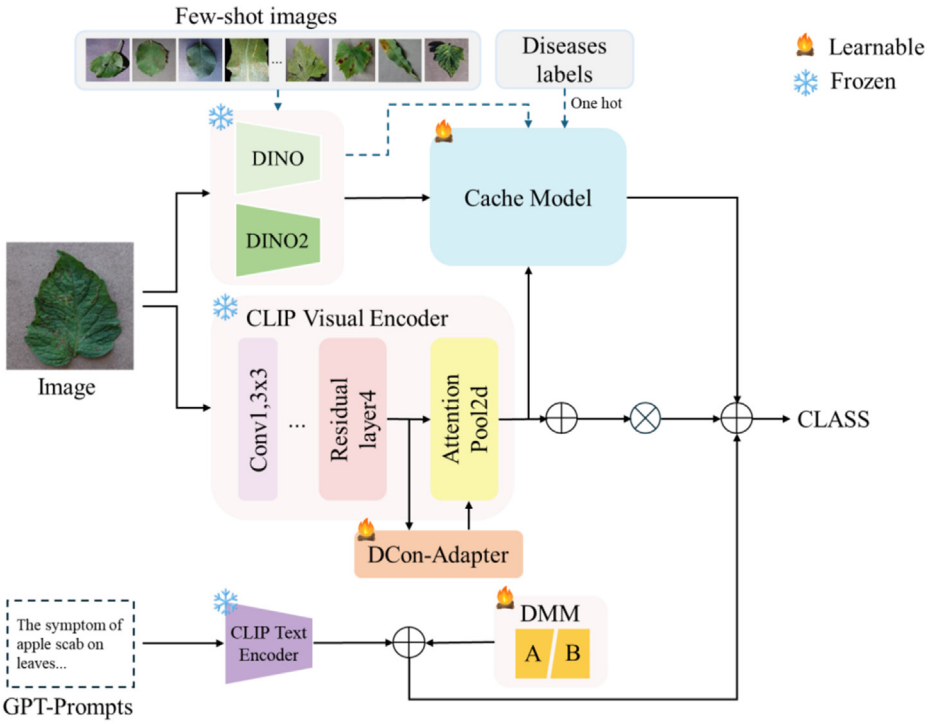


Fig. 1. The architecture of PlantCaFo. The dark blue dotted line indicates the storage of few-shot image features in advance in the buffer model.

through transformations or syntheses of existing data [11,12]. Meta-learning promotes adaptation to new tasks via training on a diverse set of tasks [13–15]. Transfer learning uses knowledge learned from a related task to assist in solving the current task [16,17]. This method has received considerable attention because it leverages information from a source domain to enhance the model's performance and generalization in the target domain.

In recent years, few-shot learning based on transfer learning for plant disease classification has typically employed a two-stage strategy: first, learning general feature representations on a large number of relevant source sets and then fine-tuning on target sets to generate specific feature representations for subsequent prediction tasks. For example, Argüesoa et al. [18] utilized the Inception V3 [19] network as a feature extractor to train on a plant disease dataset and then employed the Siamese network [20] with triplet loss to learn new plant diseases from a small dataset, achieving an average identification accuracy of 90 % for "6-way 80-shot". Hepsağ et al. [21] proposed refining a model initially trained on ImageNet [22] with PlantCLEF2022 [23], which includes nearly 4 million images across 80,000 categories, to extract embeddings. They then trained a support vector machine, yielding an accuracy of 88.4 % in a "38-way 10-shot" scenario. Li et al. [24] proposed a semisupervised few-shot classification method using a small number of labeled samples and a large number of unlabeled samples to reduce the amount of annotation work. Tassis et al. [25] cropped symptom images from the original plant disease data to assist in training. Additionally, some researchers have attempted to use transformer models for plant disease recognition [26,27]. However, these methods require a large amount of data and computational resources to train the feature extractor, and they often struggle with challenges such as class imbalance and domain shift, which hinder their generalization performance.

Inspired by the remarkable performance of foundation models such as CLIP [28] and DINO [29] in zero-shot and few-shot learning, we adopt existing large models to generate embeddings for samples in this work, thus alleviating the need for extensive data and limiting computational costs. However, existing foundation models have clear limitations in the agricultural field, such as mismatched datasets and poor generalization in agricultural scenarios, necessitating adjustments to address these issues. Cao et al. [30] proposed the ITLMLP method for cucumber disease identification, which is based on image–text–label information and integrates CLIP, self-supervised contrastive learning (SimCLR [31]), and label information. This method achieved a classification accuracy of 94.84 % on a small multimodal cucumber disease dataset. CLIP, which is trained on a large number of image–text pairs via contrastive learning, is a multimodal model with many parameters. Consequently, full fine-tuning usually leads to overfitting on certain datasets and slows down the training process. To address these challenges, several adapter-based methods have been proposed [32], which quickly adapt pretraining models to downstream tasks by introducing a few learnable parameters.

In addition to the aforementioned issues, data scarcity significantly impacts the performance of few-shot learning. Recent studies have attempted to use generative adversarial networks (GANs) [33], architectural variants, and image-to-image translation (I2I) [34] techniques for data augmentation. Bin et al. [35] introduced a GAN to generate images of grape leaf diseases. Quan et al. [36] proposed Leaf GAN, an innovative image-to-image translation system with a self-attention mechanism. However, training GANs or I2I models efficiently remains challenging. Zhang et al. [37] presented CaFo, a multibasic cascade model, which extends images by using DALL-E [38]. However, owing to the lack of agricultural domain-specific knowledge, DALL-E and other general generative models cannot produce complex details of plant diseases. Consequently, their generated images significantly differ from the actual image and therefore cannot be used directly as the desired data.

In this work, we propose an efficient few-shot plant disease recognition method based on foundation models, called PlantCaFo. This method enhances the altered CaFo [37] (a cascade model) with a dilated contextual adapter (DCon-Adapter) and a weight decomposition matrix (WDM) to efficiently improve the model's performance in plant disease recognition tasks via a small amount of data, as shown in Fig. 1. The DCon-Adapter learns new features from few-shot images to enrich image embeddings. Moreover, the WDM updates the text weights with a small number of parameters, enhancing the interaction between the text and images. These two techniques have already achieved promising results in large models [39–41], and well-designed structures can enhance the

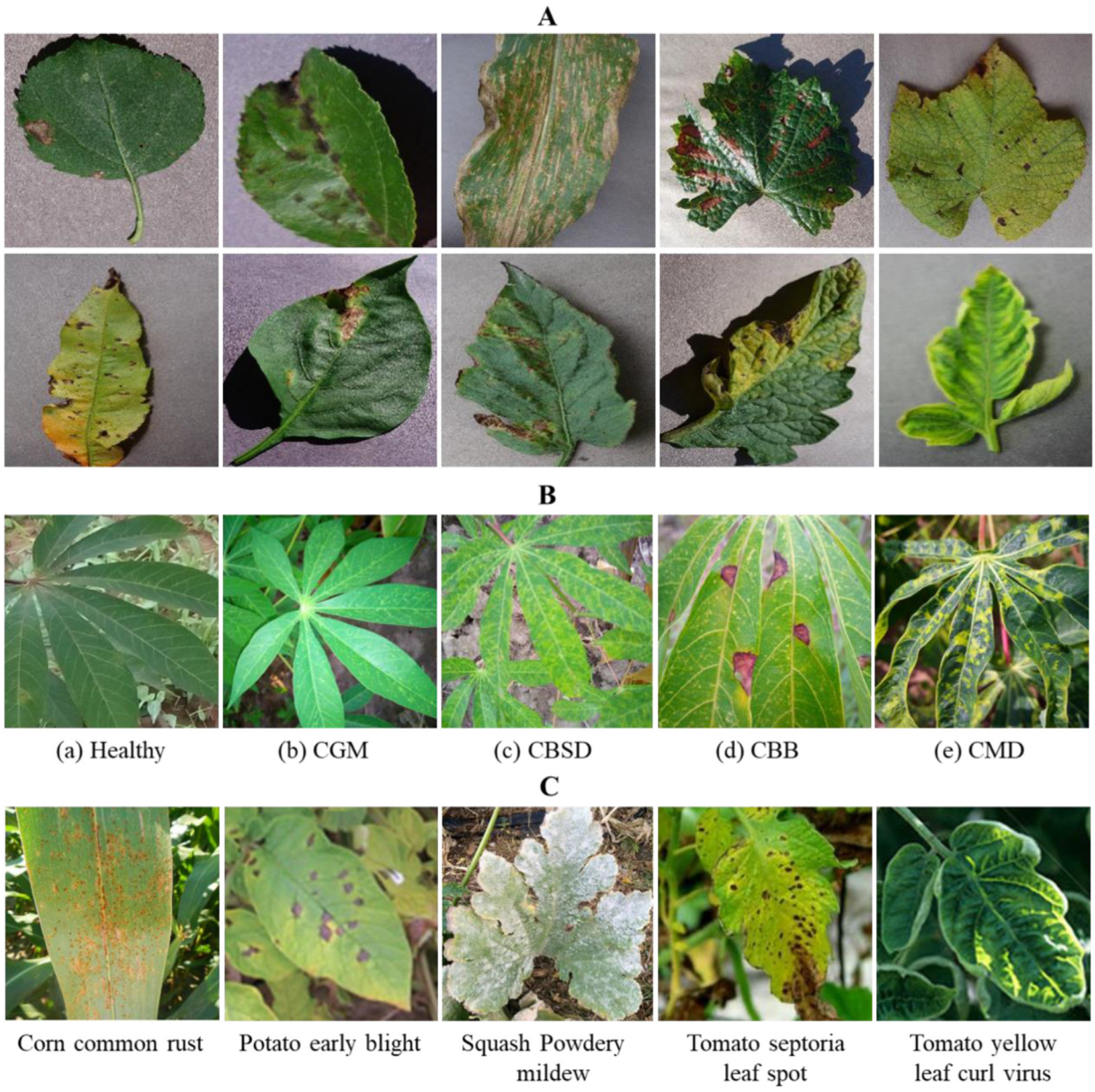


Fig. 2. Some samples from different datasets. (A) Plant Village, (B) Cassava, and (C) PDL.

model's capabilities in agricultural applications. Moreover, we explore the effects of CutMix [42] and Mixup [43] on the model. Experiments show that these two data enhancements can bridge the gap that occurs in PlantCaFo with 1 or 2 shots, further improving the model performance.

To validate the performance of our model, we conducted extensive experiments on the public Plant Village [44] and Cassava [45] datasets. The results demonstrate that our model outperforms the state-of-the-art models. Additionally, we collected samples in a natural scenario from the public PlantDoc dataset [46] to compose an out-of-distribution dataset (PDL) to assess the model's generalization ability. Our model notably enhances the accuracy on this dataset. The contributions of this work are threefold.

(1) We propose a few-shot plant disease recognition model based on foundation models that do not require pretraining in the domain.
(2) The large-scale model is fine-tuned on few-shot plant disease data via two parameter-efficient methods, a DCon-Adapter and a WDM. Extensive experiments are conducted on multiple datasets with different settings to verify the superiority of these methods.
(3) An out-of-distribution dataset (PDL) is established to validate the performance and generalization ability of the proposed methods, and PlantCaFo achieves state-of-the-art performance.

2. Materials and methods

2.1. Image datasets

We evaluate our approach on two challenging datasets: PlantVillage [44] and Cassava [45]. Additionally, we introduce a new out-of-distribution dataset (PDL) for generalization experiments. The Plant Village dataset contains 38 conditions (classes) from 14 different plants, with a total of 54,305 images collected under laboratory conditions with simple backgrounds. These plant leaves exhibit 26 disease characteristics caused by fungi, bacteria, or viruses. Some samples are shown in Fig. 2(A).

The Cassava dataset, which is from the 2019 Plant Pathology Challenge, contains 5656 images captured by smartphones in field scenarios. These images, which were collected by the Makerere University AI Lab

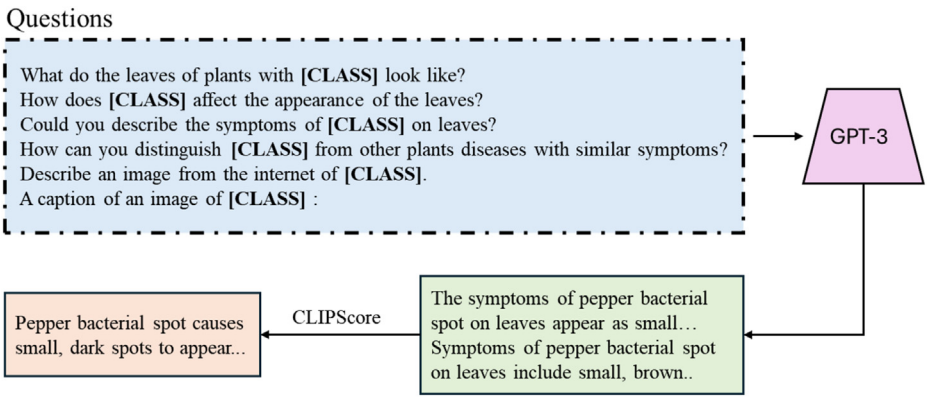


Fig. 3. The process of text generation.

Table 1
Examples of text descriptions for plant diseases.

Disease name	Text description
Apple black rot	(i) The symptoms of apple black rot are small, black spots that appear on the leaves. (ii) The symptoms of apple black rot on leaves include dark brown or black lesions on the leaves, necrosis of the tissue around the lesions, and eventually death of the leaves. (iii) Initial symptoms of black rot on apple leaves include small, dark spots that expand and eventually coalesce to form large, velvety brown lesions.
Squash powdery mildew	(i) Leaves affected by powdery mildew display white, powdery spots on their upper surfaces. (ii) The symptoms of squash powdery mildew on leaves includes the appearance of a white or gray powdery substance on the surface of the leaves. (iii) On leaves, squash powdery mildew looks like a gray or white powdery substance.

and the National Crops Resources Research Institute, are included in the training set and represent real-life scenarios. The dataset consists of five categories of cassava (four diseased, one healthy): cassava brown streak disease (CBSD), cassava mosaic disease (CMD), cassava bacterial blight (CBB), cassava green mite (CGM), and healthy cassava. Typical leaf samples are shown in Fig. 2(B). These datasets were split into training, testing, and validation sets at a ratio of 7:2:1 via stratified random sampling. Augmentation was implemented through random horizontal flipping, resizing, cropping, and normalization. The images were resized to 224 × 224 pixels to meet the requirements of the deep learning models.

To evaluate how well the model generalizes to different data distributions, we constructed a plant disease dataset (PDL) from the publicly available PlantDoc dataset [46], which captures natural scenes. This dataset has a different distribution from that of the PlantVillage dataset but shares 21 common classes, with approximately 40 images per class. Some examples from this dataset are shown in Fig. 2(C).

2.2. Disease description generation

For disease description generation, we propose two methods. First, a provided template can be used, such as “a leaf photo of [CLASS]”, where [CLASS] is replaced by the ground-truth text label for each class (e.g., “apple black spot disease” or “cucumber powdery mildew disease”). The template is used only to explore the impact of text on models. Second, disease prompts can be generated via the GPT-3 language model [47], following the process outlined in Fig. 3. To diversify the generation of disease descriptions, we design six questions with different characteristics, following the approach of CuPL [48]. These questions are suitable for all 38 different categories, with only the class name changing for each category. We then submit these questions to GPT-3 to generate 60 responses for each category. The generated texts are then filtered to retain texts related to images of the same category. We calculate the similarity between images and texts via the CLIPScore [49] to evaluate their degree of correlation. Finally, we select the top 10 texts with the highest CLIPScore for per category as the end disease description. This process not only ensures the diversity and relevance of the disease description but also provides a reliable evaluation criterion to ensure a good semantic match between the selected texts and images. Disease descriptions for some categories are shown in Table 1. Our experiments use these descriptions as prompts.

2.3. A revisit of CaFo

Extensive research has shown that large-scale pretraining can significantly enhance a network's ability to represent information, especially in regard to few-shot learning tasks. Zhang et al. [37] proposed CaFo, a Cascade of Foundation model, to explore whether multiple self-supervised models can effectively integrate prior knowledge to improve performance in few-shot recognition tasks. This model incorporates diverse prior knowledge from four pretraining paradigms, specifically, CLIP's language-contrastive knowledge, DINO's vision-contrastive knowledge, DALL-E's vision-generative knowledge, and GPT-3's language-generative knowledge. Among these, CLIP is a multimodal model trained on a large number of image–text pairs via contrastive learning. DINO is a self-supervised visual pretraining model that enhances the performance of visual tasks via contrastive learning to map different images of the same category to a shared space. DALL-E is trained on large-scale image–text pairs, which can generate images corresponding to previously unseen text descriptions. Furthermore, GPT-3 can produce coherent text based on a few manually designed prompts because of its massive text corpus training experience. By blending the strengths of these models, CaFo has demonstrated excellent performance on multiple image classification datasets compared with that of other models.

Specifically, CaFo works in a ‘prompt, generate, then cache’ pipeline. (1) Initially, manually designed prompts are input to GPT-3 to generate class-specific texts T_N with domain-specific semantics. (2) Next, these text prompts are fed into zero-shot DALL-E to automatically synthesize images. (3) CLIP selects D reliable images for each category from these synthetic images as new extended training samples. In the “N-way K-shot” setting, there are N categories, and each category contains K samples. We have a limited number of training images $I_{N,K}$, with corresponding labels $L_{N,K}$. Additionally, there are synthetic training images $I_{N,D}$, with labels $L_{N,D}$ for each class. Therefore, for each category, there are a total of $K^{'}$ training images $I_{N,K^{'}}$ as follows:

$$I_{N,K^{'}} = I_{N,K} + I_{N,D} \tag{1}$$

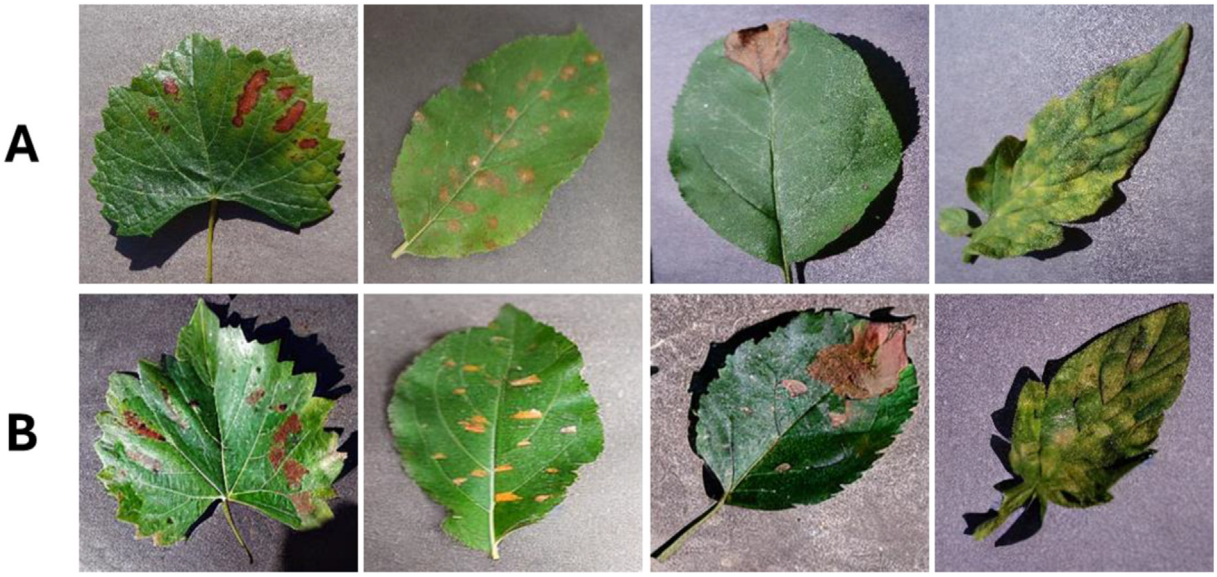


Fig. 4. Some images of plant diseases. (A) Original images and (B) generated images.

$$K^{'} = K + D \tag{2}$$

The cache model comprises visual features and their labels from the training set. For each class, the visual features $f_{D\_\text{cache}}, f_{C\_cache} \in R^{NK^{'} \times C}$ of the $K^{'}$ training images are extracted via DINO and CLIP, which serve as keys in the cache model. The corresponding labels $l_{hot} \in R^{NK^{'} \times N}$ are stored in one-hot encoding as shared values. For a test image, the process is as follows: First, CLIP (using ResNet50 as the image encoder) and DINO (with ResNet50 as the backbone network) extract its visual features $f_i \in R^{1\times C}, (i \in (DINO, CLIP_{vis}))$. The similarity with the cache values is subsequently calculated to determine the predicted score p_i. These scores are then adaptively blended with the zero-shot result $p_{zero-shot}$ of CLIP to constitute the predictions of the cache model. W_{Text} represents the prompt embeddings extracted by the text encoder of CLIP, $\phi(x) = \exp(-\alpha \cdot (1 - x))$ is a nonlinear modulator used to control the smoothness of the similarity matrix, and w_i serves as the allocated weight. The equation is as follows:

$$p_{zero-shot} = f_{CLIP} W_{Text}^{T} \tag{3}$$

$$p_i = \phi(\tag{4}$$

$$w_i = p_i \cdot p_{zero-shot} \tag{5}$$

The final prediction P is joined by the cache model and CLIP:

$$P = p_{zero-shot} + \lambda \sum_i p_i \cdot softmax(w_i) \tag{6}$$

Here, $i \in (DINO, CLIP_{vis})$, and λ is a hyperparameter. Importantly, during training, only the cache module is learnable, whereas the parameters of other modules (CLIP, DINO, etc.) are frozen. This approach of blending various pretraining knowledge sources has proven highly effective. By leveraging the strengths of multiple foundation models, CaFo demonstrates outstanding performance across a wide range of few-shot downstream classification tasks.

2.4. Details of PlantCaFo

General generative models such as DALL-E [38,50] often struggle to generate reliable images of plant diseases. Fig. 4 shows an example where we attempt to fine-tune SVDiff [51] with a small number of samples to generate plant disease images. Compared to real images, the disease lesions in the generated images are misaligned, and the coloration does not accurately reflect the typical appearance of the disease, resulting in unnatural or unrealistic depictions. The results reveal that the generated images exhibit suboptimal quality and require a substantial number of samples to produce satisfactory results. Therefore, this work adopts a modified version of CaFo as the baseline, referred to as CaFo-Base, with the image generation component removed. Additionally, Udandarao et al. [52] observed that CLIP's contrastive training maximizes the cosine similarity between paired image–text samples across modalities but neglects intramodality similarity. In light of this issue, our experiment updates the image encoder of CLIP in the cache model with the visual pretraining model DINO2 [53]. To further enhance the model's performance, we introduce two new modules: a DCon-Adapter module and a WDM module for optimization. The resulting enhanced model is called PlantCaFo, as illustrated in Fig. 1.

2.4.1. Dilated contextual adapter

CLIP is a foundation model capable of extracting general embeddings. However, its applicability is limited in agricultural applications, especially in plant disease prediction [30]. To address this limitation, we introduce a lightweight dilated contextual adapter (DCon-Adapter).

Inspired by the superiority of adapters [32,54,55], the DCon-Adapter is designed to fine-tune parameters while preventing potential overfitting in few-shot learning scenarios, which is strategically placed after the last convolutional block of the CLIP image encoder (which uses ResNet-50 [56] as its backbone). It consists of four layers: the first layer is a dilated convolution layer, which captures global features by expanding the receptive field, which is particularly useful for handling complex backgrounds in plant disease recognition tasks; the second layer is a batch normalization (BN) layer, which standardizes feature distributions to accelerate training and improve stability; the third layer uses the ReLU activation function, which introduces nonlinearity to enhance learning capacity and offers computational efficiency due to its simple derivative, accelerating the backpropagation process; and the fourth layer is a standard convolution layer, which is used to refine local features, further improving the model's classification ability in few-shot settings. Through this four-layer structure, the DCon-Adapter effectively balances the extraction of global and local features, improving the model's performance in few-shot learning, especially in cases with scarce data and complex backgrounds, leading to significant improvements in plant disease recognition accuracy. Moreover, we use residual connections to

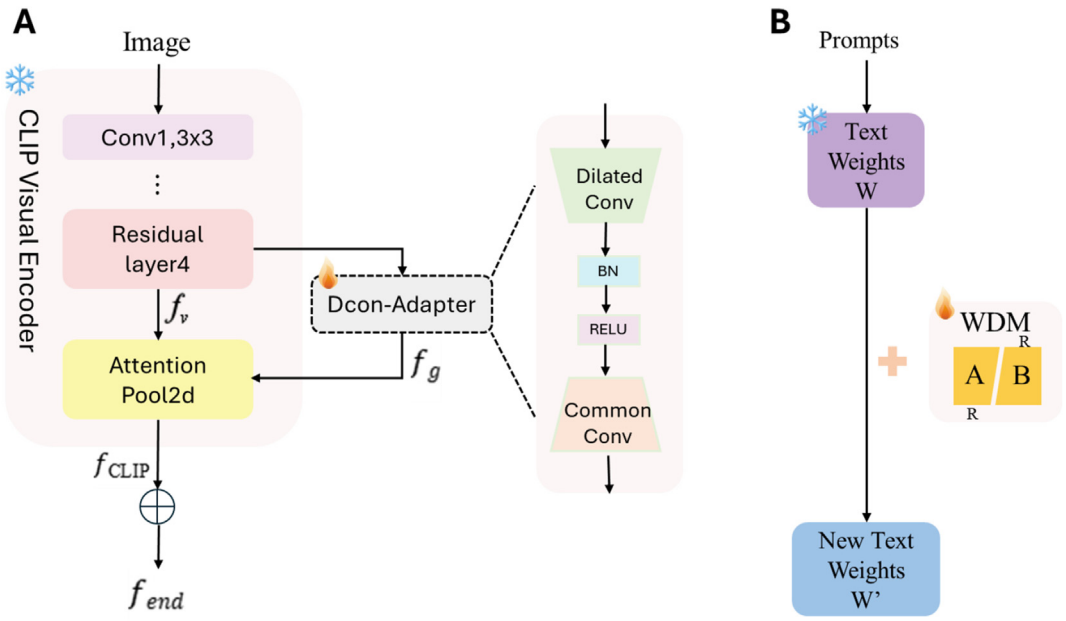


Fig. 5. Structure of the modules. (A) Design of the DCon-Adapter and (B) update of the text weight via the WDM.

blend new information learned by DCon-Adapter with pretraining prior knowledge. This approach ensures adaptation to new tasks without losing prior knowledge.

As illustrated in Fig. 5(A), the process works as follows: given an input image I, the spatial visual features $f_v \in R^{H\times W\times C'}$ are extracted by the CLIP image encoder's last convolutional block. The extracted features are processed by the DCon-Adapter, which then produces new global and local visual features, denoted as $f_g \in R^{1\times C}$. To guarantee that prior knowledge is not neglected, these new features are combined with the original features $f_{CLIP} \in R^{1\times C}$ extracted by the CLIP image encoder via a residual connection. In summary, the process of feature extraction by DCon-Adapter can be expressed as follows:

$$f_v = CLIP_{\text{vis\_layer4}}(I) \tag{7}$$

$$f_g = DCon\_adapter(f_v) \tag{8}$$

The new learned information is incorporated into the original features:

$$f_{end} = avg\left(f_{CLIP} + attention\left(f_g\right)\right) \tag{9}$$

Then, Equation (3) is updated:

$$p_{few-shot} = f_{\text{end}} W_{Text}^{T} \tag{10}$$

After the fused image features f_{end} are acquired, the similarity matrix $p_{few-shot}$ between the image and text is calculated via the updated Equation (10). Equation (6) is subsequently applied to weight the final predicted scores derived from $p_{few-shot}$ and the cache model. Finally, the *argmax* function is employed to determine the predicted image category. When the image encoder employed by CLIP is a Vision Transformer (ViT) [57], the last transformer layer of ViT can be added to the DCon-Adapter.

In summary, the DCon-Adapter effectively enhances feature learning in the image encoder of the CLIP model by incorporating dilated convolution, batch normalization, nonlinear activation, and regular convolution layers. The residual connection combines new and prior knowledge, improving the model's robustness and adaptability. These approaches significantly increase the model's performance in few-shot settings.

2.4.2. *The weight decomposition matrix*

In deep neural networks, image classification is typically achieved by multiplying the image features with the classifier weights, resulting in a score matrix. This matrix is then transformed into a probability matrix via the *SoftMax* function [58], with the class label being determined by the index corresponding to the maximum value in the matrix. In CLIP, a similarity matrix is computed between image features and text features for each class. The class label is determined by the text with the highest similarity to the image. A comparison reveals that the embeddings extracted by the text encoder function similarly to those extracted by the classifier in image classification. Therefore, the prompt embeddings can be understood as the weights of the classifier. As in image classification, the weights of the classifier can be adjusted, allowing for the prompt embeddings to be adjusted as well. Importantly, excessive parameters can lead to overfitting in few-shot tasks. To address this issue, a similar decomposition approach inspired by Hu et al. [59,60] is adopted to decompose a custom trainable matrix (M) into two low-rank matrices (A and B):

$$M = A \cdot B \tag{11}$$

The text weights are updated via M:

$$W' = W_{Text} + M \tag{12}$$

We denote the updated text weights as W', as depicted in Fig. 5(B). Therefore, the similarity matrix between images and text in CLIP after fine-tuning is calculated as follows:

$$p_{CLIP-final} = f_{end} \cdot W' \tag{13}$$

The final classification result P is updated as follows:

$$P = p_{\text{CLIP}-final} + \lambda \sum_i p_i \cdot softmax(w_i) \tag{14}$$

This procedure allows for efficient adjustment of text weights to better align with visual information from images. Ablation studies have demonstrated that the WDM significantly enhances the model's performance in few-shot plant disease recognition.

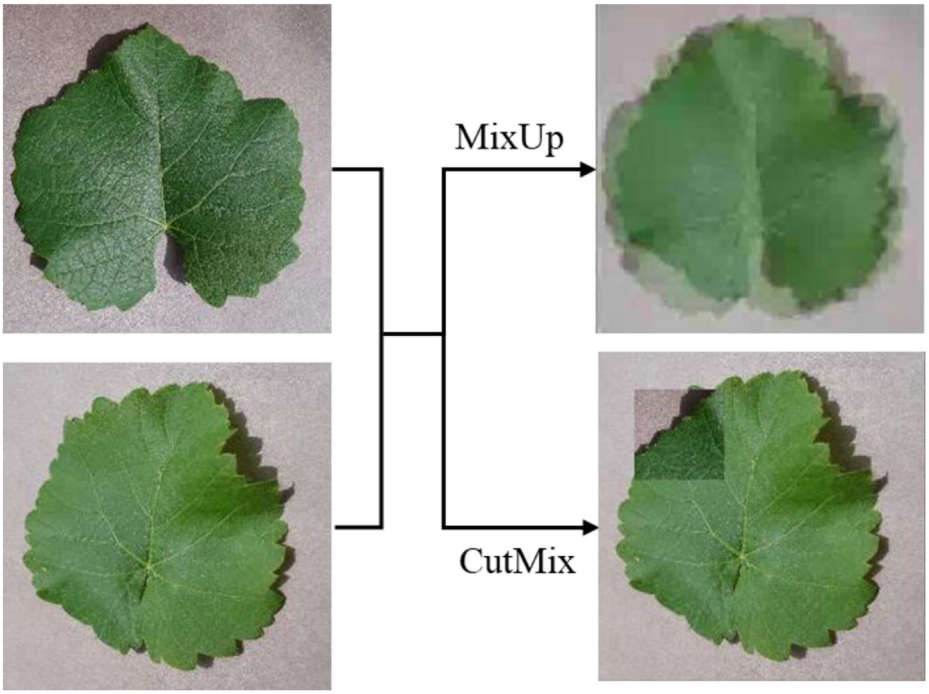


Fig. 6. Augmentation samples.

2.5. Data augmentation: Mixup and CutMix

While CaFo uses DALL-E to produce synthetic images, the common generative model DALL-E often lacks the agricultural domain-specific knowledge needed to produce complex details of plant diseases. To address this limitation, we employ the Mixup [24] and CutMix [25] data augmentation techniques to increase model performance. Some samples are shown in Fig. 6.

Mixup is a data augmentation technique based on linear interpolation. It generates new samples by linearly interpolating two different training examples in a batch, along with their labels:

$$\tilde{x} = \tau x_i + (1-\tau)x_j \tag{15}$$

$$\tilde{y} = \tau y_i + (1-\tau)y_j \tag{16}$$

where (x_i, y_i) and (x_j, y_j) are two randomly selected samples and their labels in the same batch, while $\tau \in (0, 1)$ is a number randomly sampled from the beta distribution and is used to control the interpolation weights; the default value is 0.5 in the experiment. With this method, Mixup introduces noise and disturbances to increase the robustness of the model.

CutMix involves cropping out sections of an image and randomly filling them with regions from other images in the training set. Labels are allocated proportionally. Mixup uses information from the entire image to merge two images, whereas CutMix mixes images by cropping and pasting parts of the image. Its implementation formula is similar to that of Mixup but uses a parameter to control the cropping size. CutMix requires the model to recognize objects from a local view and adds information from other samples into the cropped region, which can enhance the localization ability of the model and improve its classification performance.

2.6. Evaluation metrics

We use the average accuracy as the primary metric to evaluate the performance of few-shot classification, as this metric intuitively reflects the overall classification capability. The accuracy represents the proportion of samples correctly classified by the model and is calculated as follows:

$$Accuracy = \frac{\sum_i N_{i,i}}{\sum_{i,j} N_{i,j}} \tag{17}$$

where $N_{i,i}$ represents the number of correctly predicted images in class i and $N_{i,j}$ represents the total number of images in class i (true label) classified as class j (predicted label). Additionally, $i, j \in (1, n)$, where n represents the number of classes.

In addition to average accuracy, we plot a confusion matrix [61] to analyze the classification results in detail. The confusion matrix is an $n \times n$ matrix. Each row represents the actual class, and each column represents the predicted class. This matrix helps us visualize how well the model classifies different classes, including common misclassifications.

Furthermore, we conducted a visual analysis of the attention maps generated by the DCon-Adapter. Attention maps help us understand the key areas on which the model focuses. We used these maps to visualize the attention distribution of the DCon-Adapter on different samples, providing a more intuitive representation of the model's focus on images.

3. Results and discussion

3.1. Implementation details

In CLIP, the image encoder is ResNet-50, and the text encoder is Transformer. DINO and DINO2 use ResNet-50 and distilled ViT-S/14 as backbone networks, respectively. All the networks are pretrained. Following CaFo, we train our PlantCaFo with 1, 2, 4, 8, and 16 samples, which are randomly selected and consistent with the random seeds in the comparison experiments. During training, only the cache model, DCon-Adapter, and WDM are set to be learnable.

We train PlantCaFo and PlantCaFo*(with Mixup and CutMix data augmentations) using a batch size of 256 for 40 epochs and adopt the AdamW optimizer [62] with an initial learning rate of 0.0001. In PlantCaFo*, for augmented images, since each sample has two labels, we calculate the cross-entropy loss [63] on each label separately and weight

Fig. 7. Results of the experiments. (A) Performance (%) comparison on PlantVillage (38 conditions) and (B) performance (%) comparison on Cassava (5 conditions).

Table 2
Accuracy on PlantVillage (%). Bold indicates the best performance, and ' indicates the second-best performance.

Shots	1	2	4	8	16
Tip-Adapter-F	45.11	56.68	69.53	81.89	85.58
APE-T	52.37	64.32	73.26	80.95	86.21
CaFo-Base	62.74	73.32	78.05	87.58	92.58
PlantCaFo	62.53	72.58	82.63	89.21	93.53
PlantCaFo*	**62.95**	**73.42**	**85.11**	**91.58**	**94.23**

Table 3
Accuracy on cassava (%).

Shots	1	2	4	8	16
Tip-Adapter-F	40.06	**48.00**	**58.40**	60.40	68.00
APE-T	40.40	44.40	51.60	56.40	66.40
CaFo-Base	40.60	45.60	52.80	53.20	65.20
PlantCaFo	46.20	46.70	52.80	64.40	72.00
PlantCaFo*	**47.20**	47.60	57.60	**68.80**	**72.80**

the two losses as the final loss. Owing to the extremely unbalanced number of categories in PlantVillage, we apply the testing method by referring to other similar experiments [64]. The testing set is fixed to 50 randomly selected images from each class, and the process is repeated several times. All the hyperparameters in PlantCaFo are tuned via the official validation sets.

All the experiments use PyTorch [65] and are conducted in an Ubuntu 20.04 environment with an Intel(R) Core (TM) i5-10400 F CPU @ 2.90 GHz and an NVIDIA Tesla P40 GPU (24 GB).

3.2. Performance comparison

3.2.1. Experiments on the PlantVillage and cassava datasets

For the Plant Village (PV) and Cassava datasets, we compare PlantCaFo and PlantCaFo* (with Mixup and CutMix data augmentations) with other few-shot learning methods based on CLIP, including CaFo-Base, APE-T [66], and Tip-Adapter-F [55]. The main results are shown in Fig. 7 and Tables 2 and 3.

For the Plant Village dataset with a plain background (Table 2), Tip-Adapter-F performs best with 2 shots and 4 shots, as it has fewer parameters and shows strong generalization with fewer samples. In other

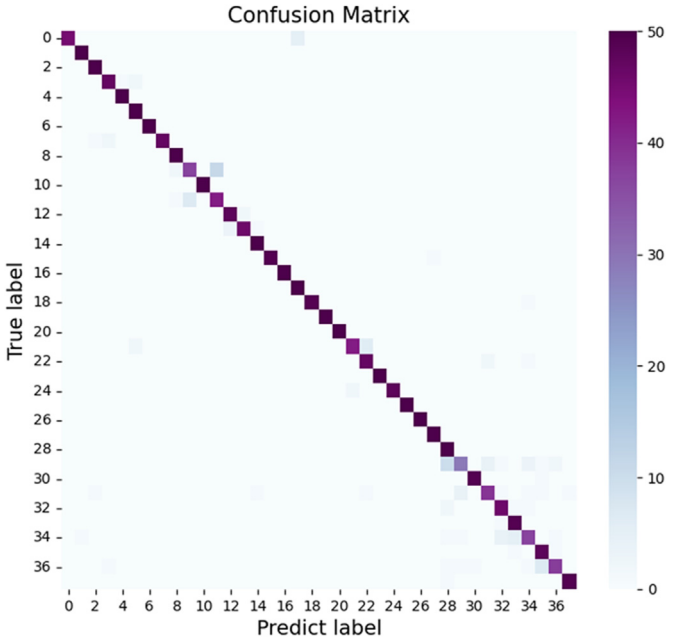


Fig. 8. Confusion matrix diagram of PlantCaFo for PlantVillage. (The confusion matrix of PlantCaFo* is similar, so it is placed in the supplementary materials.)

shots, PlantCaFo and PlantCaFo* outperform Tip-Adapter-F and APE-T, demonstrating their superior few-shot learning ability on plant disease datasets. Compared with CaFo-Base, PlantCaFo achieves competitive results with 1 or 2 samples. Notably, with more than 2 shots, PlantCaFo significantly outperforms CaFo-Base by up to 4.60 %, and PlantCaFo* shows even greater improvement. For the complex case of Cassava (Table 3), PlantCaFo and PlantCaFo* consistently outperform CaFo-Base. The confusion matrices of PlantCaFo trained with 16 samples on the test set are shown in Fig. 8. The darkest colors along the diagonal indicate high accuracy, with most predictions being correct and few misclassifications. This strong performance demonstrates PlantCaFo's ability to effectively learn and classify with limited data. The improved performance of PlantCaFo can be attributed to the use of the DCon-Adapter, which helps capture more contextual information with fewer parameters, and the WDM, which optimizes the interaction between textual and visual features. The incorporation of Mixup and CutMix augmentations

Table 4
Efficiency analysis.

Method	PlantVillage		Cassava	
	Time	Accuracy (%)	Time	Accuracy (%)
Tip-Adapter-F	2 min33 s	85.58	55 s	68.00
APE-T	2 min 34 s	86.21	1 min 4 s	66.40
CaFo-Base	6 min 10 s	92.58	1 min 35 s	65.20
PlantCaFo	13 min 47 s	93.32	1 min 57 s	70.00
PlantCaFo*	19 min 2 s	94.11	3 min 20 s	69.60

Table 5
Splits of PDL.

Split	disease name
split1	Tomato healthy, Tomato bacterial spot, Tomato early blight, Tomato late blight, Tomato leaf mold, Tomato mosaic virus, Tomato septoria leaf spot, Tomato yellow leaf curl virus
split2	Apple cedar rust, Apple scab, Corn common rust, Corn gray leaf spot, Corn northern leaf blight, Grape black rot, Grape healthy, Pepper bacterial spot, Pepper healthy, Potato early blight, Potato late blight, Raspberry healthy, Squash Powdery mildew

Table 6
Accuracy of the generalization experiment (%).

Method	PDL Split1			PDL Split2		
	8-way			13-way		
Shots	4	8	16	4	8	16
CaFo-Base	68.75	74.25	84.00	60.05	64.77	68.75
PlantCaFo	72.50	81.50	91.50	58.00	61.08	71.08
PlantCaFo*	77.00	85.00	96.50	58.31	60.77	67.38

further boosts the model's performance by enhancing its ability to generalize across different plant disease types, which likely contributes to the increased performance in the PlantCaFo* variant.

3.2.2. Efficiency analysis

Following CaFo, we evaluate the computational efficiency of each model by comparing the running time. We measure the time taken to train our model and existing methods using 16 samples on an NVIDIA Tesla P40. The results are shown in Table 4.

On the PlantVillage dataset, the runtime of PlantCaFo is slightly longer than twice that of CaFo-Base. This increase is due primarily to the large size of the PlantVillage dataset, which causes our model to spend more time extracting feature representations from the validation set. However, on the smaller Cassava dataset, PlantCaFo's runtime only increased slightly.

In summary, while our method shows a moderate increase in runtime due to the addition of learning parameters and data augmentation, the overall performance is significantly improved. Compared with the other models, PlantCaFo achieves up to 7.74 % higher accuracy, justifying the slight increase in computational cost.

3.2.3. Generalization ability

To evaluate the generalization ability of our model, we conduct experiments using an out-of-distribution dataset (PDL). We divide PDL into split1 and split2, as shown in Table 5. Split1 consists of multiple diseases from a single plant species, whereas split2 includes multiple diseases from various plant species. The models are trained on PlantVillage (source domain) with “8-way 4-shot”, “8-way 8-shot”, “8-way 16-shot”, “13-way 4-shot”, “13-way 8-shot” and “13-way 16-shot” settings and then tested on split1 and split2 of PDL. The results are presented in Table 6.

In the experiments, PlantCaFo and PlantCaFo* demonstrate strong performance on split1, which consists of various tomato diseases, indicating effective adaptation to plant diseases of the same type. However, on split2, which includes a broader range of plants, such as apple, corn, and grape diseases, our model underperforms CaFo-Base on the 4-shot and 8-shot settings. Interestingly, applying CutMix and Mixup augmentations resulted in even lower performance than without them. This performance gap can be attributed to the domain shift and the more complex backgrounds present in split2 than in the simpler PlantVillage dataset (Fig. 2A) used for training. While split1 contains diseases with relatively more consistent features, split2 introduces additional variability that poses a challenge for models trained on simpler datasets. This difference highlights the need for further adaptation to real-world agricultural scenarios, where disease symptoms may be more diverse and varied. The results suggest that although our model is effective in more controlled settings, it requires additional improvements to generalize well across a wider range of crops and environmental conditions.

3.3. Ablation studies

3.3.1. Analysis of each module

To assess the contributions of each component in our method, we conduct extensive ablation experiments on the PlantVillage dataset. The results are summarized in Table 7. A closer look at the first and second rows of Table 7 shows that DINO2 is more reliable than CLIP in computing image similarity. However, DINO2 has a smaller impact on the overall model performance improvement. Moreover, the results in the third and fourth rows indicate that the DCon-Adapter plays a more significant role than the WDM does. Namely, it supplements new knowledge effectively. The performance improvement becomes more pronounced as the number of samples increases.

With respect to the component combination, combining the DCon-Adapter and the WDM further improves the performance of few-shot learning; however, this combination may be limited when the number of samples is small (e.g., 1 or 2 samples) because of the learning ability of the trainable parameters. Finally, this issue has been effectively addressed by introducing data augmentation techniques. These results demonstrate that our method achieves significant performance in few-shot learning tasks by synergistically combining these four components: DINO2, DCon-Adapter, WDM, and data augmentation.

3.3.2. Impact of prompt design

To explore the effect of text prompts on model performance, we conduct experiments using a simple template: “a leaf photo of [CLASS]”.

Table 7
Accuracy analysis of each module on PlantVillage (%). DINO2 stands for DINO2 inside the cache model. DCon-Adapter means a dilated contextual adapter module. WDM is a weight decomposition matrix module. AG consists of Mixup and CutMix data augmentations. “-" indicates the absence of the component, whereas "+" indicates its presence.

DINO2	DCon-Adapter	WDM	AG	1-shot	2-shot	4-shot	8-shot	16-shot
-	-	-	-	62.74	73.32	78.05	87.58	92.58
+	-	-	-	62.63	73.11	80.00	87.79	92.26
+	+	-	-	60.21	71.89	81.95	88.95	93.37
+	-	+	-	62.68	73.11	80.05	88.89	92.37
+	+	+	-	62.53	72.58	82.63	89.21	93.53
+	+	+	+	62.95	73.42	85.11	91.58	93.89

Table 8
Accuracy in PlantVillage using simple prompts (%).

Shots	1	2	4	8	16
CaFo-Base	53.26	66.53	66.53	82.84	89.42
PlantCaFo	59.32	69.68	81.86	85.74	90.79
PlantCaFo*	59.42	72.26	82.63	89.26	91.84

The experimental results are presented in Table 8. The results demonstrate that PlantCaFo and PlantCaFo* consistently outperform CaFo-Base across all shot settings when simple prompts are used. The performance gap increases with the number of shots, suggesting that even with simple text prompts, our model still has better capabilities in understanding and utilizing text information.

3.4. Visualization of the DCon-Adapter

To better explain the impact of the DCon-Adapter on the model's superior performance, we use Smooth Grad CAM++ [67,68] to visualize the feature attention maps of selected images. These maps intuitively show the image pixel positions that the model focuses on when identifying plant leaf diseases. We conduct visualizations on 16 samples and compare them with those of other models.

As shown in Fig. 9, the three subfigures from left to right represent the original image, the attention map of CaFo-Base, and the attention map of PlantCaFo. The brighter the color is, the greater the contribution of that area to the classification. The second column of the figure shows that CaFo-Base focuses on the most discriminative parts of the leaf or disease features. In contrast, as evident from the third column, PlantCaFo not only attends to the disease features but also effectively filters out irrelevant features after processing by the DCon-Adapter. Importantly, however, owing to the model's ability to recognize a wide variety of plant species and diseases, the attention maps generated by PlantCaFo are not as finely tuned or specific as those generated by models trained on a single plant disease or species. This finding is particularly evident when attention maps across different plants with similar disease symptoms are compared. As shown in the attention maps of CaFo-Base and PlantCaFo for the first and second samples, PlantCaFo tends to focus on the entire plant leaf, which may include both relevant and less relevant areas. While this approach enhances the model's ability to generalize across a diverse set of plant diseases, it may result in less precise attention for individual plant species, as the model must balance multiple features from different plants.

Nonetheless, this capability is crucial for fine-grained classification tasks, such as leaf disease identification across multiple plant species. In these tasks, different plants may exhibit similar disease characteristics, making them difficult to distinguish based on disease features alone. This saliency map demonstrates the improvement achieved by the DCon-Adapter while highlighting the trade-off between specialization and generalization when diverse datasets are used.

4. Conclusion

In this study, we propose PlantCaFo, an efficient few-shot learning model for plant leaf disease identification that is based on foundation models and is specifically designed for data-limited agricultural

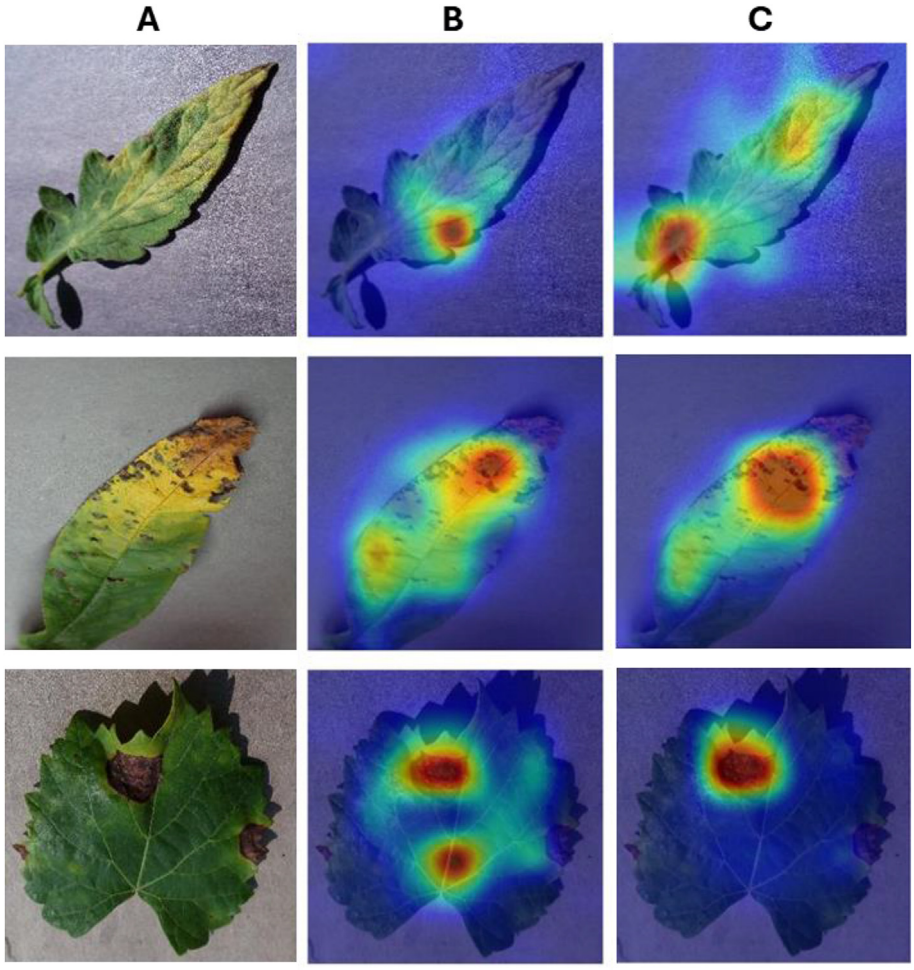


Fig. 9. Visualization of some plants via different models. (A) Original, (B) CaFo-Base, and (C) PlantCaFo.

scenarios. Our approach incorporates several key components: (1) a DCon-Adapter to enhance image feature representation, (2) a WDM to promote image-text interaction, and (3) the application of PlantCaFo and PlantCaFo* in practical scenarios demonstrates the effectiveness of the first two proposed methods. Extensive experiments demonstrate that these methods not only achieve leading results in few-shot learning but also exhibit high efficiency. Furthermore, we introduce a dataset comprising 21 categories from real-world agricultural scenarios. This dataset can serve as an out-of-distribution benchmark for future few-shot learning experiments, allowing researchers to test their model's generalization ability. This work contributes to advancing the field of agricultural AI, particularly in scenarios where data availability is limited.

However, there are certain limitations to our approach. While PlantCaFo demonstrates strong performance in controlled environments, its ability to generalize to highly diverse and complex agricultural scenarios may be limited because of the inherent challenges in handling variations in plant disease appearance and image quality. The use of the DCon-Adapter, while improving the feature extraction process, still faces difficulties in capturing all fine-grained disease patterns across different plant species. Additionally, although our approach works effectively on out-of-distribution datasets, the performance gap between different datasets, especially those with complex backgrounds or rare diseases, suggests that further improvements in model robustness are needed.

We propose several potential directions for future work: (1) **Designing hierarchical models:** For complex plant disease recognition tasks, a hierarchical model architecture can be designed to classify plants and diseases at different levels. The first layer can perform coarse classification (e.g., plant type recognition), whereas the second layer can further identify specific diseases. (2) **Designing specialized adapters for different plant disease categories:** Future work could explore the design of multiple, task-specific adapters for plant disease recognition. By categorizing plant diseases into broader groups, distinct adapters can be tailored for each category, enabling the model to learn more specialized features. This modular approach may improve the performance on diverse disease types and enhance the model's ability to generalize across different categories. (3) **Designing an adapter trained via meta-learning:** By leveraging the concept of meta-learning, an adapter that can adapt quickly to few-shot tasks can be designed. Through training on multiple tasks, the meta-learning model can learn how to adjust the adapter's parameters more effectively, thereby demonstrating stronger adaptability and generalization abilities for new plant disease tasks.

Author contributions

Xue Jiang: Resources, Conceptualization, Methodology, Formal analysis, Writing-original draft, Writing review & editing. **Jiashi Wang:** Investigation, Visualization, Writing-review & editing. **Kai Xie:** Data processing, Validation. **Chenxi Cui:** Data processing, Validation. **Aobo Du:** Data processing, Validation. **Xianglong Shi:** Data processing, Validation. **Wanneng Yang and Ruifang Zhai:** Project supervision, Writing-review & editing.

Data availability

The data used in this study are publicly available. They can be downloaded from the following websites: https://plantvillage.psu.edu/, https://www.kaggle.com/c/cassava-disease and https://github.com/pratikkayal/PlantDoc-Dataset.

Funding

This work was supported by the National Key Research and Development Program of China (2023YFF1000100).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

General: We would like to express our sincere gratitude for all contributions.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.plaphe.2025.100024.

References

[1] L.R. Brown, The World Outlook for Conventional Agriculture: more emphasis is needed on farm price policy and plant research if future world food needs are to be met, Science 158 (3801) (1967) 604–611.
[2] P.A. Nazarov, D.N. Baleev, M.I. Ivanova, L.M. Sokolova, M.V. Karakozova, Infectious plant diseases: etiology, current status, problems and prospects in plant protection, Acta naturae 12 (3) (2020) 46.
[3] F. Faithpraise, P. Birch, R. Young, J. Obu, B. Faithpraise, C. Chatwin, Automatic plant pest detection and recognition using k-means clustering algorithm and correspondence filters 4 (2) (2013) 189–199.
[4] M. Shoaib, T. Hussain, B. Shah, I. Ullah, S.M. Shah, F. Ali, S.H. Park, Deep learning-based segmentation and classification of leaf images for detection of tomato plant disease, Front. Plant Sci. 13 (2022). Article 1031748.
[5] M. Long, M. Hartley, R.J. Morris, J.K.M. Brown, Classification of wheat diseases using deep learning networks with field and glasshouse images, Plant Pathol. 72 (3) (2023) 536–547, https://doi.org/10.1111/ppa.13684.
[6] M. Belmir, W. Difallah, A. Ghazli, Plant leaf disease prediction and classification using deep learning. 2023 International Conference on Decision Aid Sciences and Applications (DASA), 2023, pp. 536–540. Paper presented at.
[7] X. Dong, K. Zhao, Q. Wang, X. Wu, Y. Huang, X. Wu, T. Zhang, Y. Dong, Y. Gao, P. Chen, PlantPAD: a platform for large-scale image phenomics analysis of disease in plant science, Nucleic Acids Res. 52 (D1) (2024) D1556–D1568.
[8] A. Parnami, M. Lee, Learning from few examples: a summary of approaches to few-shot learning, arXiv (2022), https://doi.org/10.48550/arXiv.2203.04291.
[9] J. Yang, X. Guo, Y. Li, F. Marinello, S. Ercisli, Z. Zhang, A survey of few-shot learning in smart agriculture: developments, applications, and challenges, Plant Methods 18 (1) (2022) 28, https://doi.org/10.1186/s13007-022-00866-2.
[10] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C.F. Wang, J.-B. Huang, A closer look at few-shot classification, arXiv, https://doi.org/10.48550/arXiv:1904.04232, 2019.
[11] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, Journal of big data 6 (1) (2019) 1–48.
[12] L. Perez, J. Wang, The effectiveness of data augmentation in image classification using deep learning, arXiv, https://doi.org/10.48550/arXiv.1712.04621, 2017.
[13] T. Hospedales, A. Antoniou, P. Micaelli, A. Storkey, Meta-learning in neural networks: a survey, IEEE Trans. Pattern Anal. Mach. Intell. 44 (9) (2021) 5149–5169.
[14] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J.B. Tenenbaum, H. Larochelle, R.S. Zemel, Meta-learning for semi-supervised few-shot classification, arXiv, https://doi.org/10.48550/arXiv:1803.00676, 2018.
[15] X. Wu, H. Deng, Q. Wang, L. Lei, Y. Gao, G. Hao, Meta-learning shows great potential in plant disease recognition under few available samples, Plant J. 114 (4) (2023) 767–782.
[16] K. Weiss, T.M. Khoshgoftaar, D. Wang, A survey of transfer learning, Journal of Big data 3 (2016) 1–40.
[17] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He, A comprehensive survey on transfer learning, Proc. IEEE 109 (1) (2020) 43–76.
[18] D. Argüeso, A. Picon, U. Irusta, A. Medela, M.G. San-Emeterio, A. Bereciartua, A. Alvarez-Gila, Few-Shot Learning approach for plant disease classification using images taken in the field, Comput. Electron. Agric. 175 (2020) 105542.
[19] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.
[20] G. Koch, R. Zemel, R. Salakhutdinov, Siamese neural networks for one-shot image recognition, in: ICML Deep Learning Workshop, 2015. Lille.
[21] P. Uskaner Hepsağ, Efficient plant disease identification using few-shot learning: a transfer learning approach, Multimed. Tool. Appl. 83 (20) (2023) 58293–58308, https://doi.org/10.1007/s11042-023-17824-2.
[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 248–255.
[23] H. Goëau, P. Bonnet, A. Joly, Overview of PlantCLEF 2022: image-based plant identification at global scale, in: CLEF (Working Notes), 2022, pp. 1916–1928.

[24] Y. Li, X. Chao, Semi-supervised few-shot learning approach for plant diseases recognition, Plant Methods 17 (1) (2021) 68, https://doi.org/10.1186/s13007-021-00770-1.
[25] L.M. Tassis, R.A. Krohling, Few-shot learning for biotic stress classification of coffee leaves, Artificial Intelligence in Agriculture 6 (2022) 55–67, https://doi.org/10.1016/j.aiia.2022.04.001.
[26] S.V. Nuthalapati, A. Tunga, Multi-domain few-shot learning and dataset for agricultural applications, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 2021, pp. 1399–1408.
[27] K. Zhao, X. Wu, Y. Xiao, S. Jiang, P. Yu, Y. Wang, Q. Wang, PlanText: gradually masked guidance to align image phenotypes with trait descriptions for plant disease texts, Plant Phenomics. 6 (2024) 272.
[28] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, Learning transferable visual models from natural language supervision, in: International Conference on Machine Learning, PMLR, 2021, pp. 8748–8763.
[29] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, A. Joulin, Emerging properties in self-supervised vision transformers, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 9650–9660.
[30] Y. Cao, L. Chen, Y. Yuan, G. Sun, Cucumber disease recognition with small samples using image-text-label-based multi-modal language model, Comput. Electron. Agric. 211 (2023). Article 107993.
[31] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: International Conference on Machine Learning, PMLR, 2020, pp. 1597–1607.
[32] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for NLP, in: International Conference on Machine Learning, PMLR, 2019, pp. 2790–2799.
[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Adv. Neural Inf. Process. Syst. 27 (2014).
[34] J.-Y. Zhu, T. Park, P. Isola, A.A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2223–2232.
[35] B. Liu, C. Tan, S. Li, J. He, H. Wang, A data augmentation method based on generative adversarial networks for grape leaf disease identification, IEEE Access 8 (2020) 102188–102198, https://doi.org/10.1109/access.2020.2998839.
[36] Q.H. Cap, H. Uga, S. Kagiwada, H. Iyatomi, Leafgan: an effective data augmentation method for practical plant disease diagnosis, IEEE Trans. Autom. Sci. Eng. 19 (2) (2020) 1258–1267.
[37] R. Zhang, X. Hu, B. Li, S. Huang, H. Deng, Y. Qiao, P. Gao, H. Li, Prompt, generate, then cache: cascade of foundation models makes strong few-shot learners, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 15211–15222.
[38] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, I. Sutskever, Zero-shot text-to-image generation, in: International Conference on Machine Learning, Pmlr, 2021, pp. 8821–8831.
[39] R. Wang, D. Tang, N. Duan, Z. Wei, X. Huang, G. Cao, D. Jiang, M. Zhou, K-Adapter: Infusing Knowledge into Pre-trained Models with Adapters, 2020 *arXiv preprint arXiv:2002.01808*.
[40] J. Wu, W. Ji, Y. Liu, H. Fu, M. Xu, Y. Xu, Y. Jin, Medical Sam Adapter: Adapting Segment Anything Model for Medical Image Segmentation, 2023 arXiv preprint arXiv:2304.12620.
[41] Z. Chen, Y. Duan, W. Wang, J. He, T. Lu, J. Dai, Y. Qiao, Vision Transformer Adapter for Dense Predictions, 2022 arXiv preprint arXiv:2205.08534.
[42] S. Yun, D. Han, S.J. Oh, S. Chun, J. Choe, Y. Yoo, Cutmix: regularization strategy to train strong classifiers with localizable features, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 2019, pp. 6023–6032.
[43] H. Zhang, M. Cisse, Y.N. Dauphin, D. Lopez-Paz, mixup: beyond empirical risk minimization, arXiv, https://doi.org/10.48550/arXiv:1710.09412, 2017.
[44] D. Hughes, M. Salathé, An open access repository of images on plant health to enable the development of mobile disease diagnostics, arXiv, https://doi.org/10.48550/arXiv.1511.08060, 2015.
[45] E. Mwebaze, T. Gebru, A. Frome, S. Nsumba, J. Tusubira, iCassava 2019 fine-grained visual categorization challenge, arXiv, https://doi.org/10.48550/arXiv.1908.02900, 2019.
[46] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, N. Batra, PlantDoc: a dataset for visual plant disease detection, in: Proceedings of the 7th ACM IKDD CoDS and 25th COMAD, 2020, pp. 249–253.
[47] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, Language models are few-shot learners, Adv. Neural Inf. Process. Syst. 33 (2020) 1877–1901.
[48] S. Pratt, I. Covert, R. Liu, A. Farhadi, What does a platypus look like? generating customized prompts for zero-shot image classification, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 2023, pp. 15691–15701.
[49] J. Hessel, A. Holtzman, M. Forbes, R.L. Bras, Y. Choi, Clipscore: a reference-free evaluation metric for image captioning, arXiv, https://doi.org/10.48550/arXiv.2104.08718, 2021.
[50] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, M. Chen, Hierarchical text-conditional image generation with clip latents, arXiv. 1(2):3, https://doi.org/10.48550/arXiv:2204.06125, 2022.
[51] L. Han, Y. Li, H. Zhang, P. Milanfar, D. Metaxas, F. Yang, Svdiff: compact parameter space for diffusion fine-tuning, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 7323–7334.
[52] V. Udandarao, A. Gupta, S. Albanie, Sus-x: training-free name-only transfer of vision-language models, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 2725–2736.
[53] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, Dinov2: learning robust visual features without supervision, arXiv, https://doi.org/10.48550/arXiv:2304.07193, 2023.
[54] P. Gao, S. Geng, R. Zhang, T. Ma, R. Fang, Y. Zhang, H. Li, Y. Qiao, Clip-adapter: better vision-language models with feature adapters, Int. J. Comput. Vis. 132 (2) (2024) 581–595.
[55] R. Zhang, R. Fang, W. Zhang, P. Gao, K. Li, J. Dai, Y. Qiao, H. Li, Tip-adapter: training-free clip-adapter for better vision-language modeling, arXiv, https://doi.org/10.48550/arXiv:2111.03930, 2021.
[56] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[57] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, An image is worth 16x16 words: transformers for image recognition at scale, arXiv, https://doi.org/10.48550/arXiv:2010.11929, 2020.
[58] S. Sharma, S. Sharma, A. Athaiya, Activation functions in neural networks, Data Sci. 6 (12) (2017) 310–316.
[59] E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: low-rank adaptation of large language models, arXiv, https://doi.org/10.48550/arXiv:2106.09685, 2021.
[60] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[61] J.T. Townsend, Theoretical analysis of an alphabetic confusion matrix, Percept. Psychophys. 9 (1971) 40–50.
[62] Z. Yao, A. Gholami, S. Shen, M. Mustafa, K. Keutzer, M. Mahoney, Adahessian: an adaptive second order optimizer for machine learning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 10665–10673.
[63] A. Mao, M. Mohri, Y. Zhong, Cross-entropy loss functions: theoretical analysis and applications, in: International Conference on Machine Learning, PMLR, 2023, pp. 23803–23828.
[64] Y. Huang, F. Chang, Y. Tao, Y. Zhao, L. Ma, H. Su, Few-shot learning based on Attn-CutMix and task-adaptive transformer for the recognition of cotton growth state, Comput. Electron. Agric. 202 (2022). Article 107406.
[65] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, Pytorch: an imperative style, high-performance deep learning library, Adv. Neural Inf. Process. Syst. 32 (2019).
[66] X. Zhu, R. Zhang, B. He, A. Zhou, D. Wang, B. Zhao, P. Gao, Not all features matter: enhancing few-shot clip with adaptive prior refinement, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 2023, pp. 2605–2615.
[67] D. Omeiza, S. Speakman, C. Cintas, K. Weldermariam, Smooth grad-cam++: an enhanced inference level visualization technique for deep convolutional neural network models, arXiv, https://doi.org/10.48550/arXiv:1908.01224, 2019.
[68] A. Chattopadhay, A. Sarkar, P. Howlader, V.N. Balasubramanian, Grad-cam++: generalized gradient-based visual explanations for deep convolutional networks, in: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2018, pp. 839–847.