Classification on Fine-Grained Plant Pathology Image Dataset

Zhengyang Wei
ICME
Stanford University
zywei@stanford.edu

Ranchao Yang
ICME
Stanford University
audreyrc@stanford.edu

Abstract

Plant pathology classification is a challenging Fine-Grained Visual Classification(FGVC) task due to high intra-class variation and low inter-class variation. This paper attempts to experiment with different attention blocks to improve the model performance on the plant pathology classification problem. We demonstrate the performance of the baselines and our improved models on the Plant Pathology Challenge 2020 dataset and analyze the results qualitatively and quantitatively. The improved methods achieve higher accuracy and yield more interpretable attention maps.

1. Introduction

Diagnosis of plant pathology and crop diseases is critical for disease prevention [12]. With high-accuracy plant pathology classification, farmers can take timely steps to restore plants to health and avoid the loss of yield of agricultural production. However, current human scouting for plant pathology is expensive and inefficient. In recent years, with the development of computer vision, for many image classification tasks, the accuracy achieved by computers outperforms humans. Therefore, computer-vision-based methods are promising to solve this problem. However, image-based plant pathology classification is different from common image classification problems. It is usually regarded as a fine-grained visual classification (FGVC) problem.

The task of fine-grained visual classification focuses on classifying objects into sub-categories from the same super-category, e.g., distinguishing bird species, aircraft models, car models, etc. This is different from traditional image classification tasks in that the images from the sub-categories share some common characteristics and are visually similar to each other. With the high intra-class variation and low inter-class variation [3], this task is more challenging.

For plant pathology fine-grained visual classification, high intra-class variation is caused by different ages of infected tissues, different light conditions, and genetic variations. The inter-class variation is subtle due to the fact that the images are all similar leaves of plants.

Feature-encoding methods and region localization methods are widely used to deal with FGVC tasks. And the latter is more interpretable and more consistent across non-rigid and rigid visual domains. So we focus on region localization methods in this work. The main ideas of existing region localization methods include employing detection or segmentation techniques, utilizing deep filters, and leveraging attention mechanisms. Moreover, those leveraging attention mechanisms outperform others. The development of attention mechanisms is rapid, and several powerful attention blocks are proposed to enhance classification and detection performance. Hence in this work, we study how to take advantage of region localization methods and different attention mechanisms to improve the performance of plant pathology fine-grained visual classification.

In our project, the input is an image of a leaf. We then use a CNN model with attention mechanisms to output a predicted plant pathology category.

Specifically, our contributions are the following:

1. We propose a framework that combines the region localization method with different attention mechanisms to explore their performance.
2. We conduct comparison experiments on Plant Pathology Challenge 2020 dataset [9].
3. We qualitatively and quantitatively analyze the results and visualize with attention heat maps, confusion matrices and a t-SNE map.

2. Related Work

2.1. Fine-Grained Visual Classification

A variety of methods have been proposed in the past years on fine-grained visual classification.

Feature-encoding approaches aim to extract fine-grained features by encoding higher order information on features.

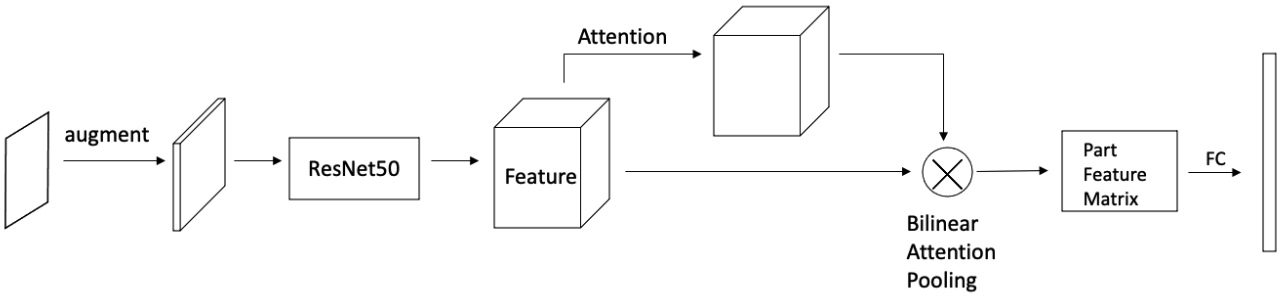


Figure 1. Model architecture.

For example, [8] computes bilinear pooled features by aggregating the pairwise interactions between features in two independent CNNs, and then computes outer product of features derived from two CNNs. Yu *et al.* [13] proposes cross-layer bilinear pooling to capture inter-layer part feature interaction. However, most end-to-end feature encoding networks are not interpretable. The accuracy across non-rigid and rigid visual domains is also not consistent.

Region localization methods have also been used widely with weakly supervised learning. They are used to detect discriminative regions in images. Weakly Supervised Data Augmentation Network (WSDAN) [7] utilizes such methods along with attention-based data augmentation and attention maps. Context-aware Attentional Pooling (CAP) [4] proposes context-aware attention-guided rich representation of integral regions to discriminate the subtle changes in an object/scene. It introduces a learnable pooling to automatically select the hidden states of a recurrent network to encode spatial arrangement and appearance features. Region localization methods might be inapplicable when the fine-grained parts are not consistent across the metacategories (e.g., iNaturalist [10]).

Recently, the meta-information (e.g., spatio-temporal prior, attribute, and text description) has been used for FGVC task. A unified and extremely effective meta-framework (MetaFormer) [14] was proposed to unify the visual appearance and various meta-information. It only used the global features but achieved state-of-the-art performance on many benchmarks. In our problem, however, we only consider utilizing the image data to realize the FGVC task due to the fact that our dataset does not contain meta-information.

2.2. Attention Mechanism

Squeeze-and-Excitation (SE) block is proposed as a novel architectural unit which explicitly models inter-dependencies between channels and adaptively recalibrates channel-wise feature responses [6]. It implicitly embeds the channel dependency through transformation, shrinks the transformation output through spatial dimensions and learns a sigmoid activation which can fully capture channel-wise dependencies. Finally, it achieves dynamic channel-wise feature recalibration and improves the model performance. The SE block is very simple and flexible so that it can be included in different architectures.

Self-attention mechanism (SAM) can also serve as a basic building block for image classification models [15]. There are two forms of self-attention, i.e. pairwise self-attention and patchwise self-attention. Both of them follow the Hadamard product forms and introduce vector attention to efficiently adapt weights across spatial and channel dimensions. The pairwise one adopts subtraction operation and is fundamentally different from convolution. The patchwise self-attention combined with the unfolding operation is a generalization of convolution. The performance of the latter one is better and is promising to yield strong accuracy gains across applications in computer vision.

Convolutional Block Attention Module (CBAM) [11] can sequentially apply the attention maps along channel dimension and spatial dimension separately, and then multiply the input feature map to refine it. As a lightweight and general module, it can be inserted into any CNN architecture seamlessly.

3. Methods

3.1. Overview

As illustrated in Fig. 1, following the structure proposed in [7], we first augment the input image using multiple data augmentation methods, and then we pass it through the ResNet50 model to obtain computed features. Then we pass the features through the patchwise self-attention block to obtain the attention map. We pass the features along with the attention map into the bilinear attention pooling block to get a part feature matrix, which is then passed through a fully-connected network to produce the probability distribution.

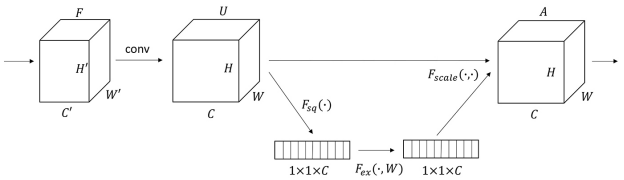


Figure 2. Squeeze and Excitation Block.

3.2. Feature and Attention Extraction

For the input image $\mathbf{I}$, we extract the features $\mathbf{F} \in \mathbb{R}^{H\times W\times C}$ with CNN. It is important to locate object's parts and extract discriminative features in them for a FGVC problem [7]. For WSDAN, the distributions of objects' parts are represented by attention maps $A \in R^{H\times W\times M}$ which are obtained from F by

$$A = f(F) = \bigcup_{k=1}^{M} A_k \tag{1}$$

where $f(\cdot)$ is a function which can output a desired attention map. In this work, we try different choices of $f(\cdot)$ to generate the attention map, including vanilla convolution functions, the squeeze-and-excitation block, the patchwise self-attention block, and the channel and spatial attention block.

3.2.1 Squeeze and Excitation Block

The Squeeze-and-Excitation block [6] is a computational unit whose input is $\mathbf{F} \in \mathbb{R}^{H'\times W'\times C'}$ and output is $\mathbf{A} \in \mathbb{R}^{H\times W\times C}$ as shown in Fig. 2.

$\mathbf{F}$ is first inputted to a convolution layer and transformed to $\mathbf{U} \in \mathbb{R}^{H\times W\times C}$ and $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_C]$, so that the channel dependencies are implicitly embedded.

Then, a channel descriptor $\mathbf{z} \in \mathbb{R}^C$ is generated by globalling spatial information of $\mathbf{U}$ according to

$$z_c = \mathbf{F}_{sq}(\mathbf{u}_c) = \frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W} u_c(i,j) \tag{2}$$

To fully capture channel-wise dependencies, a simple gating mechanism with a sigmoid activation is used to learn a flexible and non-mutually-exclusive relationship:

$$\mathbf{s} = \mathbf{F}_{ex}(\mathbf{z}, \mathbf{W}) = sigmoid(\mathbf{W}_2 ReLU(\mathbf{W}_1\mathbf{z})) \tag{3}$$

Then, transformation output $\mathbf{U}$ is rescaled with the activations $\mathbf{s}$:

$$\mathbf{a}_c = \mathbf{F}_{scale}(\mathbf{u}_c, s_c) = s_c \cdot \mathbf{u}_c \tag{4}$$

where $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_C]$.

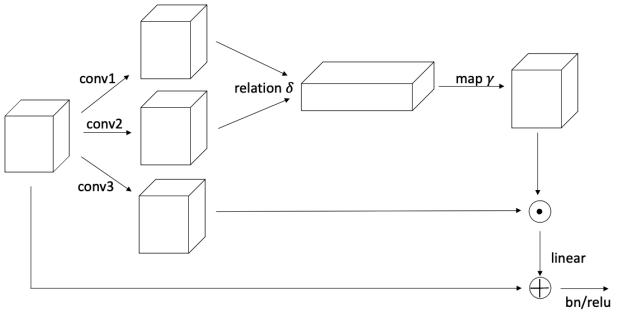


Figure 3. Patchwise self-attention block.

3.2.2 Patchwise Self-attention Block

Patchwise self-attention [15] has the following form:

$$\mathbf{y}_i = \sum_{j\in R(i)} \alpha(\mathbf{x}_{R(i)})_j \odot \beta(\mathbf{x}_j) \tag{5}$$

where $\mathbf{x}_{R(i)}$ is the patch of feature vectors from feature $\mathbf{F}$, $\alpha(\mathbf{x}_{R(i)})$ is a tensor of the same spatial dimension as the patch $\mathbf{x}_{R(i)}$, and $\alpha(\mathbf{x}_{R(i)})_j$ is the vector in this tensor corresponding spatially to the vector $\mathbf{x}_j$ in $\mathbf{x}_{R(i)}$. $\beta(\mathbf{x}_j)$ is a transformation function and $\alpha(\mathbf{x}_{R(i)})$ is computed as follows:

$$\alpha(\mathbf{x}_{R(i)}) = \gamma(\delta(\mathbf{x}_{R(i)})) \tag{6}$$

where the function δ combines the feature vectors of $\mathbf{x}_j$ and the function γ maps a vector to tensor with the unfolding operator.

The self-attention block is illustrated in Fig. 3. The feature is passed onto two streams: one computes $\alpha(\mathbf{x}_{R(i)})$, and the other one computes $\beta(\mathbf{x}_j)$. The two outputs are then aggregated via Hadamard product and passed through some basic layers. Then they are expanded to the original size of channel dimension to get the attention map $\mathbf{A}$.

3.2.3 Channel and Spatial Attention Block

Given a feature map $\mathbf{F} \in \mathbb{R}^{C\times H\times W}$ as the input, the overall process can be formulated as [11] :

$$\begin{aligned} \mathbf{F}' &= \mathbf{M}_c(\mathbf{F}) \odot \mathbf{F} \\ \mathbf{A} &= \mathbf{M}_s(\mathbf{F}') \odot \mathbf{F}' \end{aligned} \tag{7}$$

where $\mathbf{M}_c(\mathbf{F}) \in \mathbb{R}^{C\times 1\times 1}$ is a 1D channel attention map, $\mathbf{M}_s(\mathbf{F}') \in \mathbb{R}^{1\times H\times W}$ is a 2D spatial attention map, and $\mathbf{A}$ is the final refined output.

Specifically, channel attention map $\mathbf{M}_c(\mathbf{F})$ is computed as:

$$\begin{aligned} \mathbf{M}_c(\mathbf{F}) =& sigmoid(\mathbf{W}_1(ReLU(\mathbf{W}_0\mathbf{F}^{\mathbf{c}}_{\mathbf{avg}}))) + \\ & \mathbf{W}_1(ReLU(\mathbf{W}_0\mathbf{F}^{\mathbf{c}}_{\mathbf{max}})) \end{aligned} \tag{8}$$

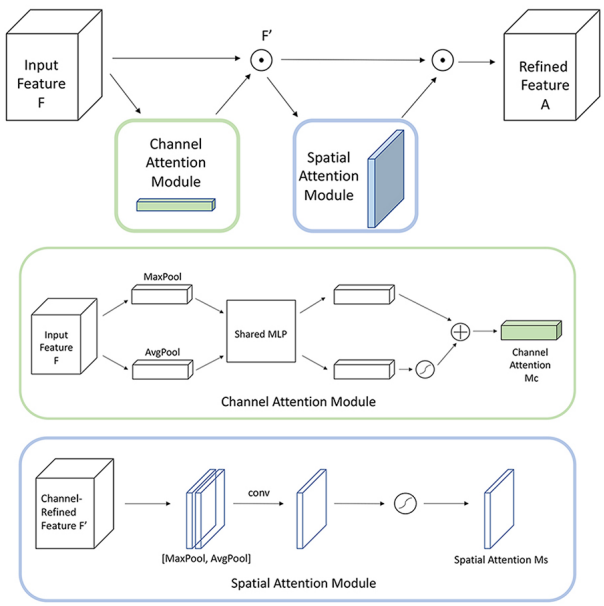


Figure 4. Channel and spatial attention modules.

where $\mathbf{X^c_{avg}} \in \mathbb{R}^{C\times1\times1}$ is the average-pooled feature and $\mathbf{X^c_{max}} \in \mathbb{R}^{C\times1\times1}$ is the max-pooled feature.

And spatial attention map is computed as:

$$\mathbf{M}_s(\mathbf{F}') = sigmoid(f^{7\times7}(\mathbf{F'^s_{avg}}; \mathbf{F'^s_{max}})) \tag{9}$$

where $\mathbf{F'^s_{avg}} \in \mathbb{R}^{1\times H\times W}$ and $\mathbf{F'^s_{max}} \in \mathbb{R}^{1\times H\times W}$ is the output of average or max pooling operation. Its detailed structure is shown in Fig. 4.

3.3. Bilinear Attention Pooling

The features of the object are represented by a part feature matrix $P \in R^{M\times N}$ which is achieved by stacking these part features f_k. Let $\Gamma(A, F)$ indicate bilinear attention pooling between an attention map A and a feature map F. It can be represented as follows:

$$P = \Gamma(A, F) = \begin{pmatrix} g(a_1 \odot F) \\ g(a_2 \odot F) \\ \cdots \\ g(a_M \odot F) \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \cdots \\ f_M \end{pmatrix} \tag{10}$$

where $g(\cdot)$ is the additional feature extraction function that can extract discriminative local features.

3.4. Training with Attention-guided Data Augmentation

In the training process, for each training image, we randomly choose one attention map A_k to guide the data augmentation, and normalize it as the k_{th} augmentation map $A^*_k \in R^{H\times W}$.

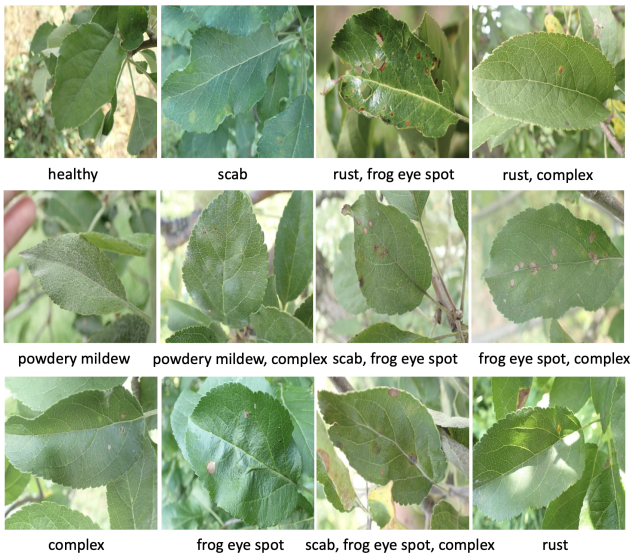


Figure 5. Examples from the dataset. Low inter-class variation.

$$A^*_k = \frac{A_k - \min(A_k)}{\max(A_k) - \min(A_k)} \tag{11}$$

Then, with augmentation map A^*_k, we can first get the crop mask C_k from A^*_k.

$$C_k(i,j) = \begin{cases} 1, & \text{if } A^*_k(i,j) > \theta_c \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

We then find a bounding box that can cover the whole selected positive region of C_k. We enlarge this region from the raw image and use it as our augmented input data. We can also obtain attention drop mask D_k from A^*_k:

$$D_k(i,j) = \begin{cases} 0, & \text{if } A^*_k(i,j) > \theta_d \\ 1, & \text{otherwise} \end{cases} \tag{13}$$

The k^{th} part region will be dropped by masking image I with D_k. Attention cropping allows the model to extract more fine-grained features, and attention dropping encourages the model to detect other discriminative parts, so that the model can perform better.

3.5. Test with Object Localization and Refinement

In the testing process, after the model outputs the coarse-stage classification result and the corresponding attention maps for the raw image, we can predict the entire region. We then enlarge it to predict the fine-grained result using the same network model.

Figure 6. Examples of leaves that are "healthy". High intra-class variation.

4. Dataset

We use Plant Pathology Challenge 2020 dataset [9] as our dataset. It consists of 18632 images of plant leaves classified into 12 categories. The resolution of the images is 448×448. We split the dataset into training, validation and test sets. Table 1 provides detailed information about our dataset.

We perform data augmentation on the training data. Our data augmentation methods include randomly cropping, randomly horizontal flipping (with the default probability parameter 0.5), randomly changing brightness (with brightness parameter 0.126), and randomly changing saturation (with saturation parameter 0.5).

Examples of the test images are provided in Fig. 5. There is low inter-class variation since apparently all the images are mostly green and have similar shapes regardless of which categories they fall in. Some of the leaves look extremely alike: for example, it is difficult to distinguish between the leaf with "complex" disease symptoms and the one with "rust" disease.

On the other hand, there is high intra-class variation in the images. Examples of images that are classified as "healthy" are shown in Fig. 6. It is obvious that the "healthy" leaves have different shapes and different backgrounds. They are also exposed to different light conditions: the third leaf is in the shade while the fourth leaf grows in the sunlight.

5. Experiments

5.1. Baselines

ResNet50 [5]. ResNet50 model is a convolutional neural network (CNN) that is 50 layers deep. Instead of learning unreferenced functions, it reformulates the layers as learning residual functions with reference to the layer inputs.

WSDAN [7] . It first generates attention maps to represent the object's distinctive parts through weakly supervised learning. Next, guided by the attention maps including attention cropping and attention dropping, it augments the image.

Category	Train	Valid	Test
scab, frog eye leaf spot	493	55	138
rust	1339	149	372
complex	1153	128	321
frog eye leaf spot	2290	254	637
rust, frog eye leaf spot	86	10	24
healthy	3329	370	925
scab, frog eye leaf spot complex	144	16	40
scab	3474	386	966
powdery mildew, complex	62	7	18
frog eye leaf spot, complex	118	14	33
rust, complex	69	8	20
powdery mildew	852	95	237

Table 1. Dataset information. Shows the number of images in the training, validation, and test sets.

5.2. Implementation Details

For WSDAN, we adapt the code provided by [2] for our own use. Here, we use SGD optimizer with the following parameters: momentum 0.9, learning rate 1e-3 and weight decay 1e-5. We train the model for 30 epochs with batch size of 12 using PyTorch on our training dataset. The channel dimension of attention map is set as 32.

We develop three variants of the WSDAN model. We add a squeeze-and-excitation block, self-attention mechanism, and a convolutional block attention module to WSDAN and denote the three models as WSDAN + SE , WSDAN + SAM , and WSDAN + CBAM respectively. For the SE block, the number of squeeze channels is set as 16. For the SAM block, the number of shared planes is set as 8. For CBAM, the ratio of channel attention is set as 1. Our code is modified based on the code provided by [2]. We added the code for three attention blocks to the model architectures and modified the corresponding training and test code. Besides, we also wrote the code for dataloader, evaluation, and visualization of the results.

Our baseline model fine-tunes pretrained ResNet-50 on our training dataset. We adapt the code from [1] and set the batch size as 32. We use the SGD optimizer, with momentum 0.9 and initial learning rate 0.01, which decays exponentially with gamma 0.975. Our model is fine-tuned for 30 epochs using PyTorch and is evaluated on the test dataset every epoch.

To avoid overfitting, we evaluate the validation data at the end of each epoch and save the models with the highest validation accuracy.

5.3. Metrics

We use accuracy to evaluate the performance of fine-grained visual classification model on our dataset. It is com-

Model	Backbone	Accuracy
Resnet50	Resnet50	88.34
WSDAN	Resnet50	89.79
WSDAN + SE	Resnet50	89.84
WSDAN + SAM	Resnet50	**90.03**
WSDAN + CBAM	Resnet50	89.90

Table 2. Accuracy comparison.

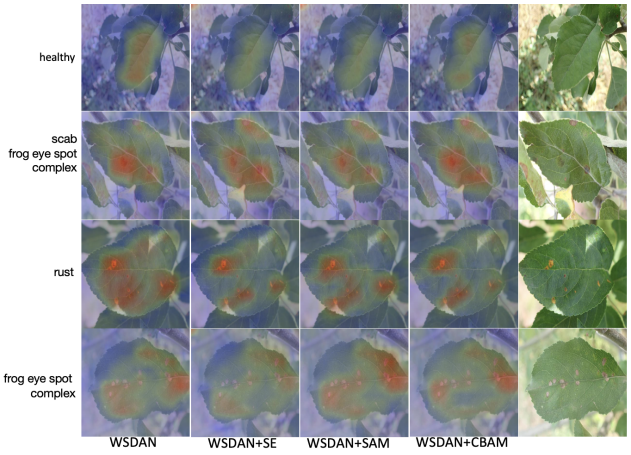


Figure 7. Examples of attention maps for each of the four models. The rightmost column is the original image.

puted as:

$$Accuracy = \frac{TP}{TP + TN} \times 100\% \tag{14}$$

where TP is number of true positive samples and TN is the number of true negative samples.

5.4. Quantitative Results

The accuracy of our baseline models and improved models is shown in Table 2.

Compared with ResNet50 model, WSDAN models achieve higher accuracy. The improvement can be attributed to the generated attention maps and attention-guided data augmentation of WSDAN models. All of improved models (WSDAN + SE, WSDAN + SAM and WSDAN + CBAM) outperform vanilla WSDAN model because of the added attention blocks. Among these three models, WSDAN + SAM obtains the highest accuracy, which implies that the self-attention mechanism is the most helpful attention mechanism for this problem.

5.5. Attention Map Visualization

Examples of the attention maps extracted using different methods are visualized in Fig. 7. For each method, we

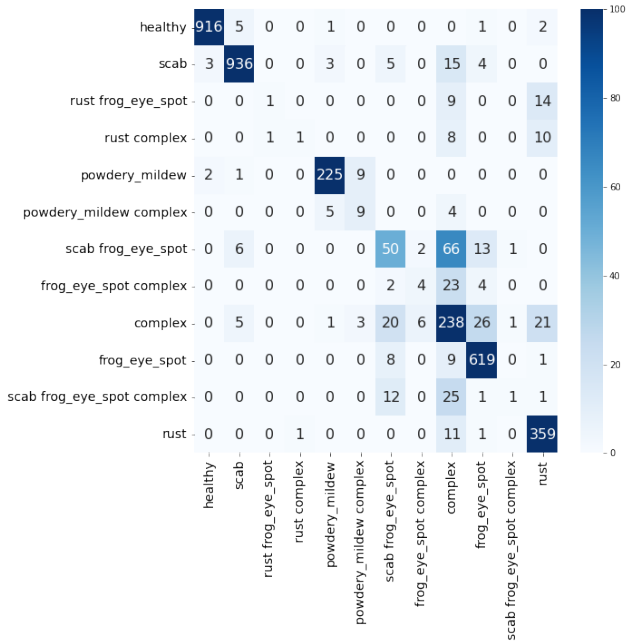


Figure 8. Confusion Matrix of WSDAN + SAM

visualize the attention maps for four images that are classified as ”healthy”, ”scab, frog eye spot, complex”, ”rust” and ”frog eye spot, complex”, respectively. The red regions in the maps represent where the attention weights are very high and the blue regions represent where attention is lower. Ideally we would expect the regions with spots to be red and the regions without spots to be blue.

We can see from the figure that WSDAN performs well since it is able to correctly detect the spots on the leaves. However, it sometimes pays too much attention to regions that are healthy. For example, there are regions in the ”healthy” and ”rust” images that are free of spots but are red.

WSDAN+SE and WSDAN+CBAM manage to achieve some improvements. For the ”healthy” image, the attention areas are less obvious and much smaller as expected. For the image classified as ”scab, frog eye spot, complex”, the models pay more attention to the upper right corner than WSDAN, where there do exist signs of disease. For the ”rust” image, the models no longer pay excess attention to healthy regions. However, the black spot at the center of the ”rust” image seems to be ”ignored” by the two models.

WSDAN+SAM yields the best attention maps among the four models. It is able to detect more precise locations of the spots for all four images. Unlike WSDAN+SE and WSDAN+CBAM, in the image classified as ”rust”, the black spot at the center is also colored as red.

(a) ”scab, frog eye spot” misclassified into ”complex”

(b) ”complex” misclassified into ”scab, frog eye spot”

Figure 9. Failure cases

5.6. Confusion Matrices

The confusion matrix of WSDAN + SAM is shown in Fig. 8 and the confusion matrices of WSDAN, WSDAN + SE, WSDAN + CBSM are appended in Appendix B Fig. B.1.

Compared with WSDAN, models integrated with additional attention blocks mainly improve the accuracy for the categories with a large number of training data. For the some categories with only a few training data, the involvement of additional attention blocks seems unable to enhance its accuracy. This might be due to the fact that insufficient training data of certain categories prevent the model from learning correct attention maps.

The overall patterns of the confusion matrices are consistent, so here we choose to focus on the confusion matrix of WSDAN + SAM to analyze the false positive results and false negative results.

There are mainly two failure cases. The first one is that the ”rust, frog eye spot” and ”rust, complex” images are likely to be classified into ”rust” images. One reason is that the number of training image of the ”rust” class is ten times more than that of ”rust, frog eye spot” and ”rust, complex”. Due to the imbalance in the training data, the model tends to classify those images into ”rust”.

The second failure case is related to the category ”complex”. Both false positive rate and false negative rate of ”complex” images are very high. Especially, more than a half of ”scab, frog eye spot” samples are classified into the ”complex” category.

Such failures might be due to the fact that ”complex” disease means more than one disease on the same leaf [9], so the spots on leaves might be of different colors and shapes. As shown in Fig. 9, the intra-class variation of ”complex” images is very high: some of them have white spots, some have black spots, and some have a combination of spots of different colors. Many of the images belonging to ”complex” category are similar to those belonging to ”scab, frog eye spot” category for human visual recognition. Hence, it might be hard for our model to detect the differences between those categories.

5.7. T-SNE Analysis

We perform t-distributed stochastic neighbor embedding (T-SNE) on the test features obtained using WSDAN-SAM which achieves the highest accuracy. Using this method, we visualize the high-dimensional features in a two-dimensional map, which is shown in Appendix A Fig. A.1. From the plot, it is obvious that the ”healthy”, ”scab”, ”powdery mildew”, and ”rust” classes are well separated from all other classes. However, the ”scab, frog eye spot”, ”frog eye spot, complex”, ”complex”, and ”frog eye spot” classes are all mixed together on the right side of the plot. This indicates that WSDAN-SAM fails to distinguish among these four classes. This result is consistent with the confusion matrix shown in Fig. 8, in which the misclassification rate is high among these classes (lower right corner) compared to other classes.

6. Conclusion

In this study, we investigate models to predict plant diseases based on images of the leaves. We apply ResNet50 and WSDAN as baselines to our dataset and experiment with three different attention blocks. We evaluate their contribution to our model improvement using accuracy as the evaluation metric. We find that WSDAN+SAM performs the best on our dataset and achieves the highest accuracy.

One limitation of this work is that, the results mentioned in earlier sections might be sub-optimal since there was not enough time to tune all the hyper-parameters. Also, another drawback is that, our dataset is highly imbalanced: some of the classes have a large number of training data while others have only a few.

In the future, we would like to adjust for the data imbalance through resampling methods. We would also like to optimize the models we have experimented with and explore other models such as transformers which might improve model performance on fine-grained visual classification tasks.

7. Contribution

We performed data exploration and the experiments together. Both contributed equally to research and literary analysis.

A. T-SNE analysis

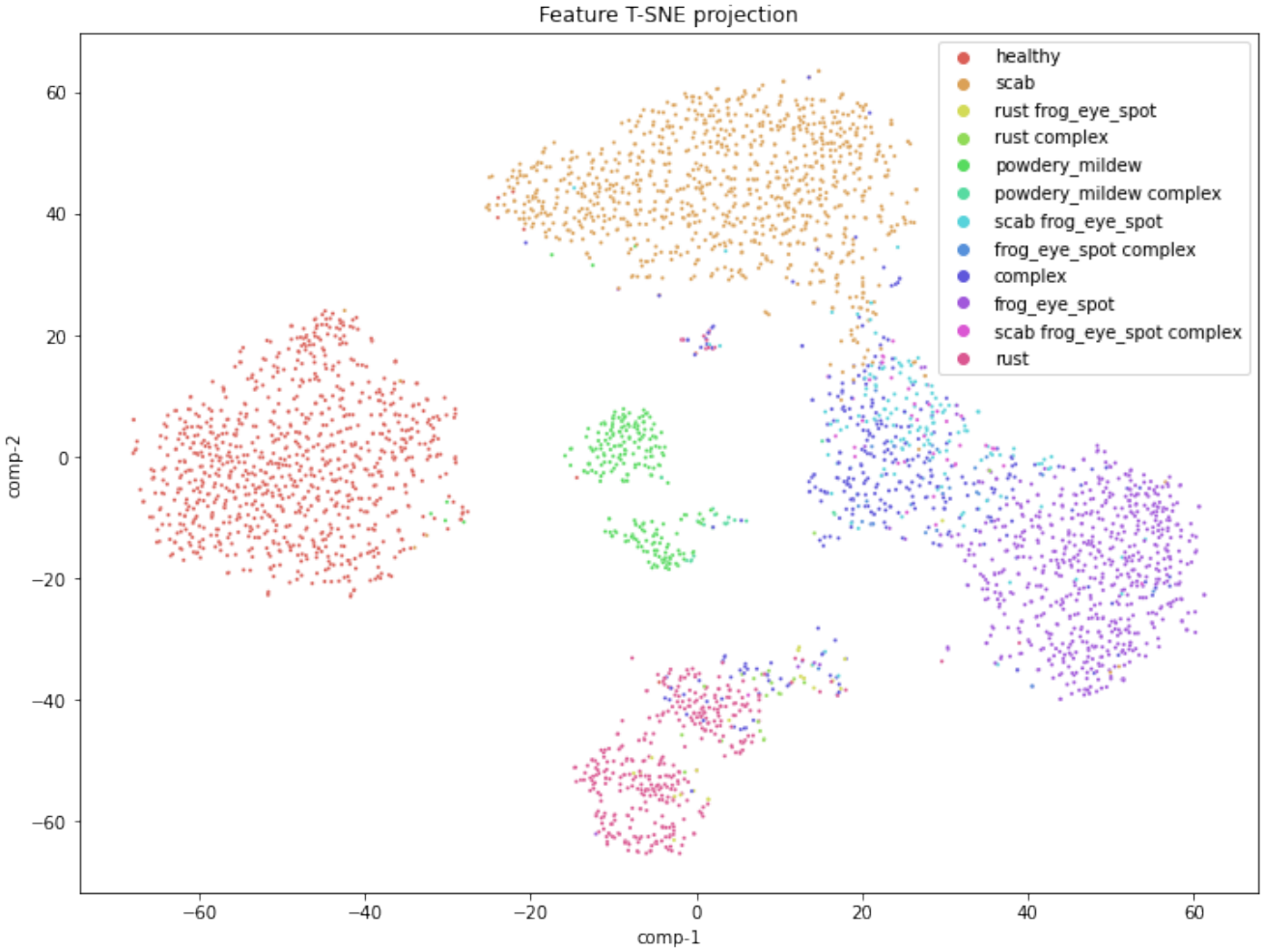
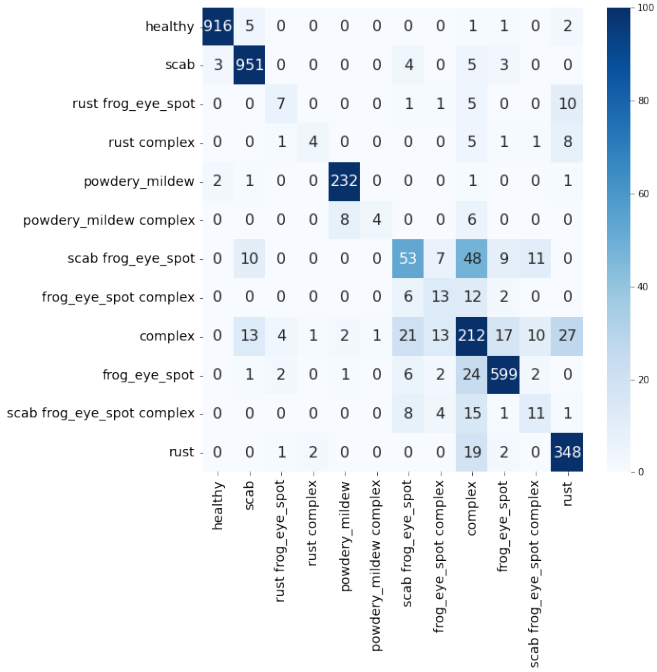
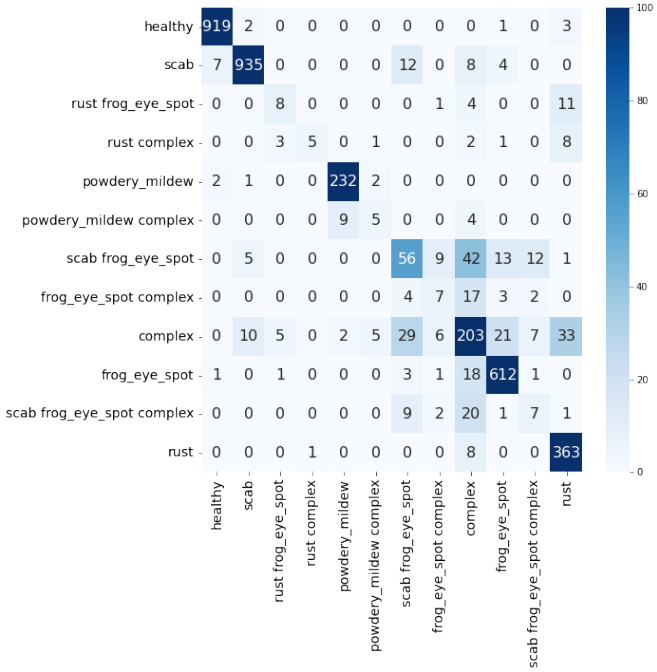


Figure A.1. T-SNE analysis.

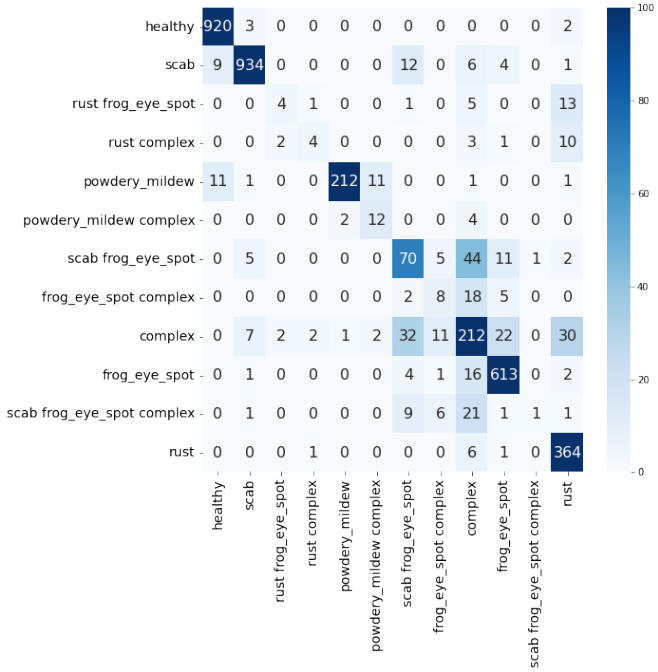
B. Confusion Matrices



(a) WSDAN



(b) WSDAN + SE



(c) WSDAN + CBAM

Figure B.1. Confusion Matrices

References

[1] Fine-tune pretrained convolutional neural networks with pytorch. https://github.com/creafz/pytorch-cnn-finetune. Accessed: 2022-05-07. 5

[2] Ws-dan.pytorch. https://github.com/GuYuc/WS-DAN.PyTorch. Accessed: 2022-05-07. 5

[3] Connor Anderson, Matthew Gwilliam, Adam Teuscher, Andrew Merrill, and Ryan Farrell. Facing the hard problems in FGVC. *CoRR*, abs/2006.13190, 2020. 1

[4] Ardhendu Behera, Zachary Wharton, Pradeep Hewage, and Asish Bera. Context-aware attentional pooling (CAP) for fine-grained visual classification. *CoRR*, abs/2101.06635, 2021. 2

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 5

[6] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2017. 2, 3

[7] Tao Hu and Honggang Qi. See better before looking closer: Weakly supervised data augmentation network for fine-grained visual classification. *CoRR*, abs/1901.09891, 2019. 2, 3, 5

[8] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear CNN models for fine-grained visual recognition. *CoRR*, abs/1504.07889, 2015. 2

[9] Ranjita Thapa, Kai Zhang, Noah Snavely, Serge Belongie, and Awais Khan. The plant pathology challenge 2020 data set to classify foliar disease of apples. *Applications in Plant Sciences*, 8(9):e11390, 2020. 1, 5, 7

[10] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset, 2017. 2

[11] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module, 2018. 2, 3

[12] Guofeng Yang, Yong He, Yong Yang, and Beibei Xu. Fine-grained image classification for crop disease based on attention mechanism. *Frontiers in Plant Science*, 11, 2020. 1

[13] Chaojian Yu, Xinyi Zhao, Qi Zheng, Peng Zhang, and Xinge You. Hierarchical bilinear pooling for fine-grained visual recognition. *CoRR*, abs/1807.09915, 2018. 2

[14] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision, 2021. 2

[15] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition, 2020. 2, 3