ZeySed, A Deep Convolutional Neural Network for Detection of Plant Diseases

Anonymous CVPR 2021 submission

Paper ID ****

Abstract

Various plant diseases can lead to a great decrease in crop production, which severely jeopardize primary human needs. Minimizing crop damage by early detection of plant diseases is required for food security of the 7 billion people on the earth. Traditional disease detection methods (e.g., naked-eye observation) are time consuming and subjective. Models that are developed based on convolutional neural network (CNN) are ideal for detecting plant diseases at their earliest stage before epidemic. To fulfill this aim, a CNN model (i.e., ZeySed) with custom architecture is developed that can successfully classify plant diseases. ZeySed is carefully tuned by course grid search to find an acceptable range of hyperparameters and then by fine random search to determine the best values of hyperparameters. The effect of each hyper parameter on the performance of model and the limitations of a relatively poor model is discussed in the paper to provide the required insight for developing a good CNN model. The accuracy of ZeySed for validation and test dataset is equal to 80.0% and 80.6%, respectively. In addition to the custom architecture, ResNet-34 was trained as baseline model for evaluating the performance of ZeySed. The comparison indicates that the accuracy of ZeySed is only 6.6% lower than ResNet-34, which looks acceptable. More precisely, the ResNet-34 takes advantage from residual learning that allows training of very deep networks. However, a very deep model is not a viable option in plain networks. This fact justifies the 6.6% difference between the accuracy of models. The proposed CNN model from this study can be used in a mobile or windows application for dynamic plant disease detection and real-time monitoring on large fields.

1. Introduction

Plant diseases are one of the essential factors that jeopardize the productions of agriculture, which are directly related to primary human needs. Detection of plant diseases at their early stages helps cure the plants and prevent disease outbreaks. Powdery and rust plant disease are two of the most common diseases around the world [2]. Figure 1 shows sample images from healthy and diseased plants. Powdery mildew is a fungus that affects many types of plants. Widespread studies were conducted in powdery mildew with the aim of disease resistance [5]. In 2003, the epidemic of soybean rust was observed in more than 90% of Brazil's field, which had more than $487 million costs for the US. [15]. The best way of detecting this disease is by looking at the plant as it leaves light grey or white powdery spots on plants. Leaf rust disease is common in humid conditions.

The plant diseases (i.e., powdery mildew and leaf rust) that are investigated in this study are highly contagious by wind or water [5, 15]. Thus, detecting these diseases at their early stages is vital for preventing from plant disease epidemic. Creating a classifier that detects the existence and type of diseases in plants helps to fulfill this aim. To shed light on this, images can be taken from huge fields in short time intervals (i.e., once a week). Then, the captured photos can be fed to the classifier to detect whether plants have disease or not. This method can prevent plant epidemic diseases that cause loss of millions of dollars. There are different methods of capturing images from plants at short time intervals. For example, a plane can fly at low altitude above the farm and take high quality photos that are proper for classifier.

This study aims to classify plant leaves into three groups of (i) healthy, (ii) powdery mildew, and (iii) leaf rust. Images were taken from different plants are used to train and test the model. We propose ZeySed, a customized/tuned neural network that classifies the plant leaves images into these three classes.

2. Related work

Application of CNN for plant disease classification has drawn attention of many researchers and many studies have been conducted in recent years [10, 7, 11, 3, 14, 12, 9]. The main incentive for conducting these studies is detecting plant diseases at its early stages. Agriculture productions can be greatly diminished owing to different diseases, which jeopardize food security. Detecting plant disease by

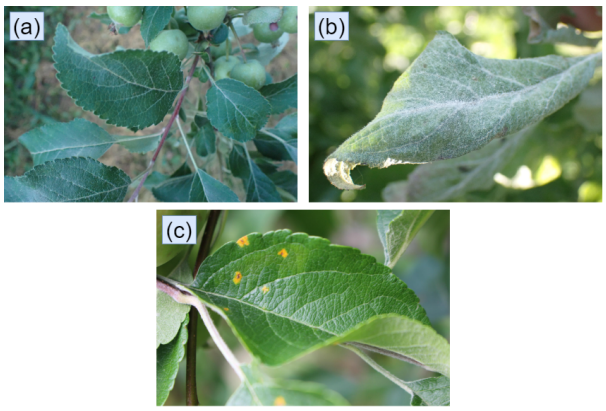

Figure 1. Images of healthy and leaves with disease (a) healthy plant; (b) powdery mildew detected by light grey color of leaves; (c) leaf rust detected by orange to brown spots

traditional methods such as naked-eye detection or laboratory test have many drawbacks including being subjective and time-consuming [9]. Thus, deep learning methods, especially CNN models are extremely helpful in early detection of plant diseases.

Ferentinos [3] utilized five CNN architectures for plant disease classification, namely, (i) AlexNet[8]; (ii) AlexNetOWTBn; (iii) GoogLeNet[13]; (iv) Overfeat; and (v) VGG. He used an openly available database of 87,848 images taken in both laboratory and cultivation field condition. The dataset contained 25 plant species in 58 distinct classes of [plant, disease] combinations. The VGG architecture achieved the highest accuracy among other architectures with the accuracy of 99.53% for test dataset. One of the drawbacks of plant disease models is that the accuracy of model reduces significantly when tested on independent data from another source. To solve this issue, Sharma et al. [11] employed segmented images to train CNN rather than using full images. They indicated that the performance of model trained with segmented images is twice better than the model trained with full images when tested on independent data. Karthiik et al. [7] employed residual learning on CNN architecture to classify 3 diseases of tomato leaves. In another study on tomato leaves, Sardogan et al. [10] employed CNN model and Learning Vector Quantization to classify four different diseases on the dataset that contained 500 images.

3. Dataset

We use *Plant Disease* dataset, a public dataset of plant images shared in Kaggle.com [1]. The dataset includes 1530 images of plants from different categories labeled into three classes: 1) Healthy, 2) Powdery, and 3) Rust. The dataset

[1]https://www.kaggle.com/rashikrahmanpritom/plant-disease-recognition-dataset

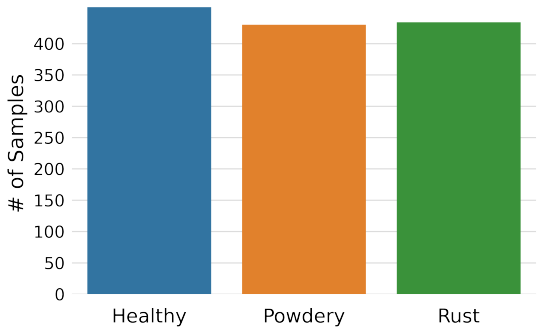

Figure 2. Distribution of different classes among training samples

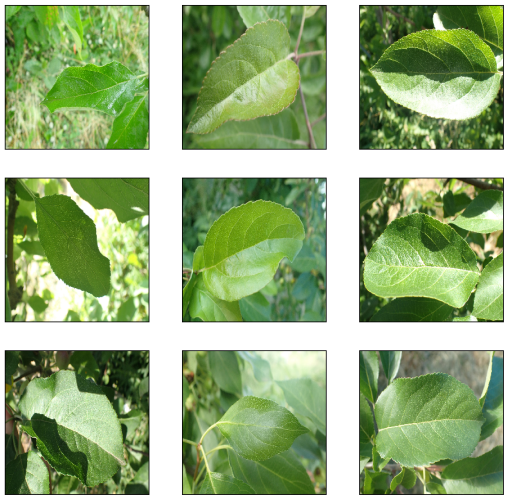

Figure 3. Random samples from Healthy class

is divided into the train (1320 samples), validation (60 samples), and test (150 samples) subsections. In Figure 2 the distribution of different classes has been depicted for train dataset. It should be noted that test dataset has the same number of images (i.e., 50) for each class. Also, to better understand the differences between the classes, we have provided random samples from each class in Figures 3, 4, and 5

4. Approach

We used deep CNNs to classify the images into categories of healthy, powdery mildew, and leaf rust. To fulfill this aim, we utilized various architecture and tuned hyper-parameters such as leaning-rate and weight regularization. For hyper-parameter tuning, we used a coarse grid to find the range in which the model's performance was higher than our minimum threshold. After narrowing the range of hyper-parameters, we employed a random search to find the

Figure 4. Random samples from Powdery class

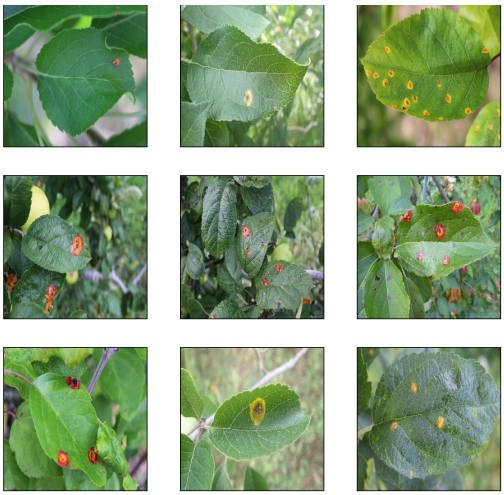

Figure 5. Random samples from Rust class

best hyper-parameters [1]. The advantage of random search over grid search is that in a random search, more points are chosen along the hyper-parameter that has a higher effect on the model performance.

The custom architecture that is proposed in this research (i.e., `ZeySed`) was tuned considering the following factors:

- **Convolutional layers**: the depth of CNNs has a major impact on the model's performance. The filters that detect light gray boundaries were useful for classifying powdery mildew, and filters that detect orange spots helped to classify rust leaves.

- **Max-pooling**: the spatial size of extracted features from convolutional layers is reduced by Max-pooling layers. As the size of images in this study was very large, Max-pooling was a great approach to reduce the dimension of the image at each convolutional layer.

- **Padding**: padding is employed in CNNs architecture to prevent reducing the dimension of each layers' input. Although we believe that padding is not required in this study due to the large size of images, we considered it as one of the factors to build the best custom architecture.

- **Batch normalization**: employing batch normalization at each convolutional layer has a significant impact on the model's performance. Batch normalization makes the model more robust to lousy initialization and increases the speed of convergence [6]. It also allowed us to use higher learning rates and simplified the creation of deep networks.

- **Dropout**: dropout prevents the over-fitting of models. When the major features (e.g., light gray color for powdery mildew and orange spots for leaf rust) were absent in some images, the model with dropout was able to classify them correctly using non-essential features. Dropout was applied on fully connected layers because they have greater number of parameters compared to convolutional layers, which might result in overfitting.

- **Fully connected layers**: after extracting features from the images by convolutional layers, fully connected layers was used at the end of architecture to correlate the features with the labels of classes. The number of fully connected layers was considered for proposing the best custom architecture.

5. Experiment

This section provides the results of plant disease classification using CNN. It is notable that accuracy was employed to evaluate the performance of model as this is a classification problem. Results are presented in both quantitative (e.g., confusion matrices, accuracy, and loss) and qualitative format (e.g., misclassified images)

We use accuracy as our proposed classification model's evaluation metric. Human eyes detect powdery mildew by seeing light grey colors of the leaves and detect rust by seeing orange to brown spots on plants (see Figure 1). Filters in CNN layers that catch the light grey color of leaves or orange spots will improve the model's accuracy by detecting powdery mildew and leaf rust, respectively. We report the accuracy and confusion matrix of the tuned model against the provided test data. Also, as a baseline classifier, we compare the performance of our model with ResNet-34 [4]. The comparison between custom architecture (i.e., `ZeySed`) and ResNet-34 indicates how far `ZeySed` is from the ideal architecture. The ResNet is easy to optimize

compared to plain networks with the same number of parameters, especially at deep models [4]. More precisely, deep networks have numerous parameters that make them very hard to optimize. Thus, increasing the depth of networks in deep architectures did not even increase the accuracy of the training dataset due to this problem.

5.1. Experimental Setup

To tune ZeySed we used NVidia 2070 super with 8GB of GPU Ram. We did a combination of grid search and random search to find the optimal values for the learning-rate and regularization factor. We explored learning-rate values between 10^{-6} and 10^{-3} and regularization factor have been selected from the values between 10^{-4} and 5×10^{-2}. Finally, the selected values for learning-rate was 1.14×10^{-5}, and 2.92×10^{-3} for the regularization factor. The average runtime for training each CNN model was between 5-6 hours on GPU. The relative high value of runtime was due to the large size of images in the dataset. Also, we used batch size of 26 for our training steps.

5.2. Results and Discussion

This section gives more details about the implementation of the method and achieved results. Figure 6 shows the training and validation accuracy of our relatively tuned model for plant disease classification. This model uses three convolutional layers and two fully connected layers; however, the final version of ZeySed uses three fully-connected layers. Also, the learning-rate and other hyperparameters did not get tuned during training this naive model. Understanding the problems of the trained model from acc-epoch figure is vitally important in creating CNN model as it provides the required insight for improving the model. In other words, if we can understand the limitations of model just by looking at the acc-epoch figure, we can directly fix the model instead of changing hyperparameters without any knowledge. As illustrated in Figure 6, this model suffers from overfitting; the training accuracy achieves 100% while the validation accuracy is 72% at epoch 20. This problem can be fixed by increasing weight regularization or adding dropout\increasing dropout ratio in FC layers. The other problem of this model is the fluctuations of accuracy. This could be attributed to the high learning rate. The validation accuracy increases very fast, achieve a relatively high value at epoch three, and starts to fluctuate, which indicates a high learning rate. More precisely, the high value of the learning-rate does not allow the model to find the global minima and reduce the loss.

Regarding the discussion above, we fully tuned ZeySed and its architecture is depicted in Figure 7. It uses three convolutional and three fully-connected layers. After each CNN layer, we performed batch normalization and gave the output to the max-pooling layer (details of model is presented in Table 1). Figure 8 shows the training and validation accuracy of fully tuned ZeySed as a function of epochs. The problems of the previous model were solved by choosing proper architecture and hyperparameters.

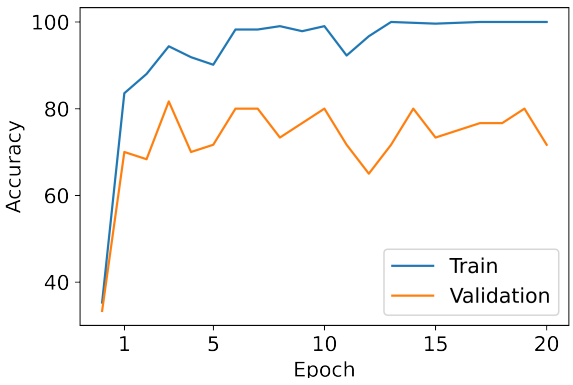

Figure 6. The training and validation accuracy of relatively tuned model as a function of epochs. Two problems of model can be extracted from this figure: (i) overfitting: the high difference between accuracy of train and validation is the indicator of overfitting and can be solved by increasing weight regularization or adding dropout\increasing dropout ratio in FC layers. (ii) fluctuations of accuracy: the accuracy of model increases very fast and fluctuates afterwards. This could be attributed to high value of learning rate

Figure 9 shows the loss of ZeySed vs epoch. The trend of loss reduction is proper (i.e., nigher too fast and nor slow) and there are no fluctuations at the final epochs. There is an increase in loss at epoch 6, but the accuracy does not decrease at this epoch. The loss increment at this epoch is due to the regularization loss (i.e., the weight of model in this epoch should be larger than adjacent epochs), and is not attributed to the data loss.

In addition to ZeySed, which is a tuned CNN model with custom architecture, ResNet-34 model was trained with the dataset to classify plant diseases. The accuracy of ResNet-34 for validation dataset was 86.6%. The ResNet-34 was used as a baseline classifier to evaluate the performance of ZeySed. The comparison between the accuracy of ResNet-34 and ZeySed, which are 86.6% and 80.0% respectively, indicates that ZeySed did a good job in classifying plant diseases. To shed light on this, ResNet-34 uses residual learning that provides the opportunity to train very deep networks with many convolutional layers. However, training very deep networks with high accuracy is almost impossible in plain network. Thus, the 6.6% difference between the accuracy of ZeySed and ResNet-34 is acceptable and indicates that ZeySed has a good performance.

The confusion matrix of validation dataset is shown in Figure 10. Among different classes, the CNN model could predict powdery better than the other two classes. In some

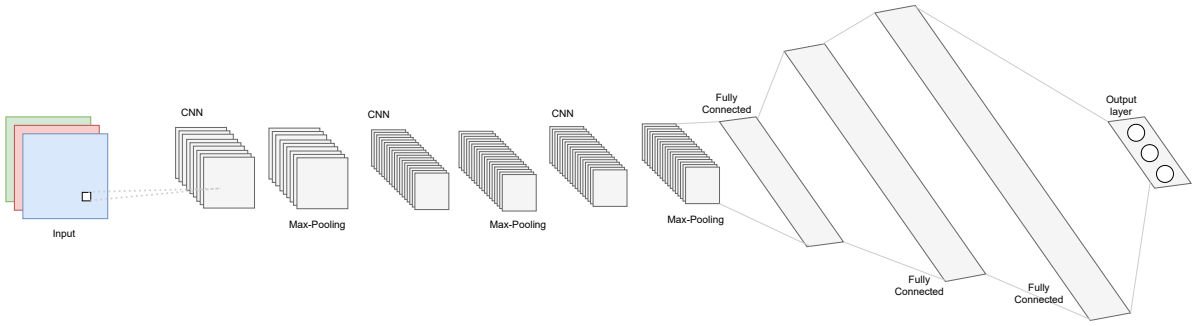

Figure 7. The architecture of ZeySed

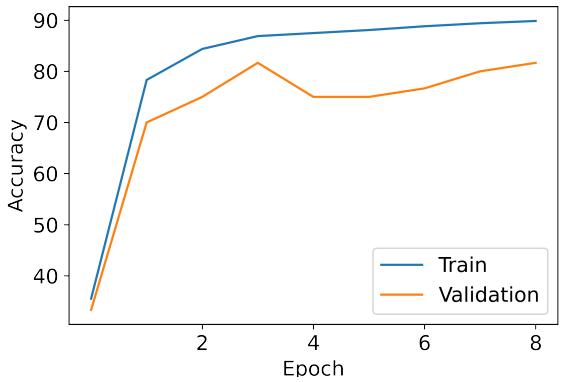

Figure 8. The training and validation accuracy of tuned ZeySed vs epochs. The problems of overfitting and acc fluctuations were solved by choosing proper architecture and hyperparameters

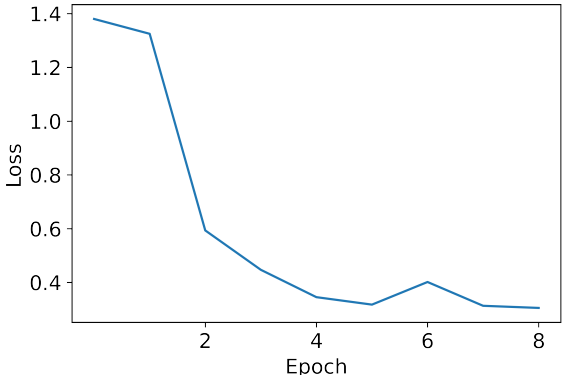

Figure 9. The loss of ZeySed as a function of epochs

Figure 10. The confusion matrix of validation dataset; powdery class was classified better than the other two classes

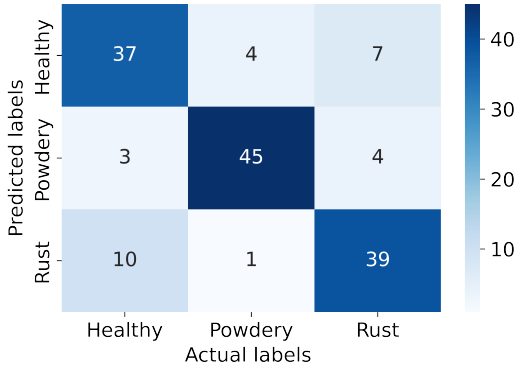

Figure 11. The confusion matrix of test dataset; the same pattern of accuracy as validation dataset is seen for different classes

cases, healthy plants are wrongly classified to rust and vice versa. A comprehensive discussion around the reasons for misclassification of images is provided in the next section. Fig 11 shows the confusion matrix of test dataset. The pattern of accuracy for different classes is similar to the validation dataset, which is rational. The accuracy of test dataset is 80.6%, which is slightly higher than validation accuracy.

Layer		
Input	out size	256 x 256 x 3
CNN1	input size	256 x 256 x 3
	out size	256 x 256 x 32
	kernel size	3 x 3
	padding	1
	activation function	ReLu
MaxPool1	input size	256 x 256 x 32
	out size	256 x 256 x 32
	kernel size	3 x 3
	padding	1
CNN2	input size	256 x 256 x 32
	out size	256 x 256 x 64
	kernel size	3 x 3
	padding	2
	activation function	ReLu
MaxPool2	input size	256 x 256 x 64
	out size	128 x 128 x 64
	kernel size	2 x 2
	padding	0
CNN3	input size	128 x 128 x 64
	out size	128 x 128 x 128
	kernel size	3 x 3
	padding	2
	activation function	ReLu
MaxPool3	input size	64 x 64 x 128
	out size	64 x 64 x 128
	kernel size	2 x 2
	padding	0
Flatten	input size	64 x 64 x 128
	out size	524,288
FC1	input size	524,288
	out size	256
	activation function	ReLU
FC2	input size	256
	out size	64
	Activation function	ReLU
FC3	input size	64
	out size	3
	Activation function	Tanh
	Dropout p	0.5

Table 1. Architecture of ZeySed

5.3. Effect of Batch Normalization and Dropout on Performance of ZeySed

In this section, we investigate the effect of two important techniques, batch normalization and dropout, on the performance of ZeySed. To this end, we modified ZeySed to see how much its performance changes. First, we removed the batch normalization from convolutional layers. Although ZeySed is not a deep neural network, we expect that removing the batch normalization would decrease its performance. We removed the batch normalization layers during training the modified ZeySed, and all other hyper-parameters and model-parameters remained the same. In Figure 12 the training and validation accuracy of modified ZeySed has been plotted. We can see that the validation accuracy of ZeySed decreased significantly. Also, more fluctuations appeared in the pattern of the accuracy vs. Epochs. The validation accuracy of ZeySed without batch normalization was near 71%, which is about 9% lower than the validation accuracy of tuned ZeySed.

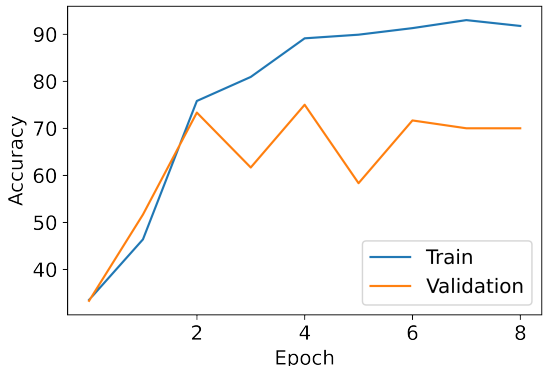

Figure 12. Training and validation accuracy of modified ZeySed. Batch normalization has been removed from convolutional layers. The validation accuracy decreased 9%. Also, the fluctuation in validation accuracy appeared again.

Then, we investigate the effect of dropout on the performance of ZeySed. The fully tuned ZeySed uses the dropout layer just in the final fully-connected layer (FC3). To see how much the overuse of dropout can impact the performance of ZeySed, we added dropout layers with $p = 0.1$ before convolutional layers (each node gets zero weight with a probability of 0.1), and we added dropout layers with $p = 0.5$ before each fully connected layers. All other hyper-parameters and model-parameters remain the same. The performance of this version of ZeySed is depicted in Figure 13. We can see that the training and validation accuracy both decreases since dropout is an example of regularization. Nevertheless, the validation accuracy of the model after 8 epochs was near 70%, 10% lower than the performance of fully tuned ZeySed.

5.4. Miss Classification

In this section, we discuss the cases that ZeySed could not predict the actual label. One of the most challenging aspects of this project was the dataset. The images of leaves have been taken in different conditions, like under direct sunlight or from different angles. This makes the classification problem more challenging since the critical regions

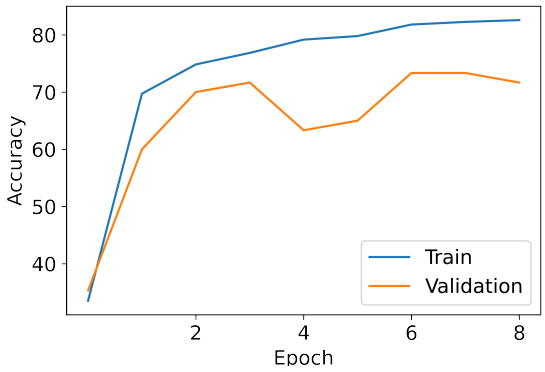

Figure 13. Training and validation accuracy of modified ZeySed. Dropout with $p = 0.1$ has been added to each convolutional layer, and dropout with $p = 0.$ has been added to each fully-connected layer. The validation accuracy decreased 10%.

of classes (like red/yellow spots in Rust leaves) might be unclear or covered by another leaf. In Figure 14 we have plot three images that ZeySed did the wrong classification for them. The image (a) of Figure 14 corresponds to the Healthy leaves, while ZeySed classified it as Rust leaf. We can see that there is bright sunlight in the image, which makes the fraction of the leaf very bright and yellow. In this case, if the model has not seen lots of Healthy samples under direct sunlight, it might get fooled since it considers a yellow spot on leaf and classifies it as Rust leaf. Figure 14 (b) shows an image labeled as Rust leaves. There are multiple leaves in this image where all except one are Healthy, and only one of them is Rust. Also, because the target leaf is not necessarily in the center of the image, it is predictable that classification of this image is a challenging task, even for humans. Lastly, image (c) of Figure 14 shows an image labeled as Healthy while it has some white spots on which makes it very similar to the Powdery leaves.

6. Conclusion

Plant diseases epidemic can completely ruin the crop production and causes loss of millions of dollars as shown in [15]. The best approach for controlling plant diseases epidemic is detecting diseases at their early stage. In this study, we have proposed a CNN classifier that detects plant diseases from leaf images. ZeySed is the CNN model with custom architecture. To tune hyperparameters of ZeySed, coarse grid search was employed to determine the range in which the model has acceptable performance. Then, fine random search was used to determine the best values for hyperparameters. The insight that was extracted from a relatively poor model, had a major impact in efficiently improving the performance of ZeySed. The accuracy of ZeySed for validation and test dataset is equal to 80.0% and 80.6%, respectively. In addition to the custom architecture, ResNet-34 was trained as a baseline classifier for comparison purposes. The accuracy of ResNet-34 for validation dataset is 86.6%. Although the accuracy of ResNet-34 is 6.6% higher than ZeySed, the performance of ZeySed is very good with respect to the facts that it does not use residual learning and therefore it has limitations in depth of model. More importantly, the ZeySed is much simpler than the ResNet-34. It uses a fewer number of parameters which reduces the cost of training and testing significantly.

Plotting misclassified examples indicated the reasons for errors in the model. The bright sunlight changes the color of leaves and makes the image extremely hard for classification even for humans. The other reason is existence of multiple leaves with different classes in one image. In these cases, ZeySed cannot correctly classify the image as there are multiple classes.

The proposed CNN model requires very low computation for classifying a given image. The low required computation makes the model feasible to be used in a mobile application by farmers and growers. The images can be taken at short time intervals by drones, planes, or agricultural vehicles. This method provides the opportunity for dynamic disease detection and real-time monitoring in large fields.

7. Future Work

The first potential research is training a CNN model for plant disease detection on a larger dataset that contains more diseases and plant types. A richer dataset enables the model to classify different type of plant disease that might have different procedures for curing. Another important suggestion about the dataset is that it should contain images both from greenhouses and fields. The limited dataset that only has images from one of the sites, will have reduced accuracy on the other.

Another potential work is coding a mobile or windows application that uses CNN model to classify plant diseases. Then, it suggests the required actions for curing the plant with respect to the type of disease and type of plant.

References

[1] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012. 3

[2] Kathryne L Everts, Steven Leath, and Patrick L Finney. Impact of powdery mildew and leaf rust on milling and baking quality of soft red winter wheat. *Plant disease*, 85(4):423–429, 2001. 1

[3] Konstantinos P Ferentinos. Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145:311–318, 2018. 1, 2

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceed-*

(a) (b) (c)

Figure 14. Three samples that ZeySed could not predict the actual label. The image (a) corresponds to the healthy leaf, while ZeySed classified it as Rust leaf. The bright sunlight spot makes the fraction of the leaf very bright and yellow, which is very similar to the Rust leaves. Image (b) shows Rust leaves. There are multiple leaves in this image where all except one are healthy, and only one of them is Rust. ZeySed classified this image as Healthy. Predictably, the classification of this image is a challenging task, even for humans. Image (c) is a Healthy labeled leaf while it has some white spots on which makes it very similar to the Powdery leaves.

ings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016. 3, 4

[5] Ralph Hückelhoven and Ralph Panstruga. Cell biology of the plant–powdery mildew interaction. *Current opinion in plant biology*, 14(6):738–746, 2011. 1

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 3

[7] R Karthik, M Hariharan, Sundar Anand, Priyanka Mathikshara, Annie Johnson, and R Menaka. Attention embedded residual cnn for disease detection in tomato leaves. *Applied Soft Computing*, 86:105933, 2020. 1, 2

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 2

[9] Jinzhu Lu, Lijuan Tan, and Huanyu Jiang. Review on convolutional neural network (cnn) applied to plant leaf disease classification. *Agriculture*, 11(8):707, 2021. 1, 2

[10] Melike Sardogan, Adem Tuncer, and Yunus Ozen. Plant leaf disease detection and classification based on cnn with lvq algorithm. In *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, pages 382–385. IEEE, 2018. 1, 2

[11] Parul Sharma, Yash Paul Singh Berwal, and Wiqas Ghai. Performance analysis of deep learning cnn models for disease detection in plants using image segmentation. *Information Processing in Agriculture*, 7(4):566–574, 2020. 1, 2

[12] Garima Shrestha, Majolica Das, Naiwrita Dey, et al. Plant disease detection using cnn. In *2020 IEEE Applied Signal Processing Conference (ASPCON)*, pages 109–113. IEEE, 2020. 1

[13] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2

[14] S Yegneshwar Yadhav, T Senthilkumar, S Jayanthy, and J Judeson Antony Kovilpillai. Plant disease detection and classification using cnn model with optimized activation function. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pages 564–569. IEEE, 2020. 1

[15] JT Yorinori, WM Paiva, RD Frederick, LM Costamilan, PF Bertagnolli, GE Hartman, CV Godoy, and J Nunes Jr. Epidemics of soybean rust (phakopsora pachyrhizi) in brazil and paraguay from 2001 to 2003. *Plant Disease*, 89(6):675–677, 2005. 1, 7