CS230

Convolutional Neural Networks to Detect Plant Disease in Common Food Crops

Colin Shi
Department of Mechanical Engineering
Stanford University
cshi100@stanford.edu

Eric Zeng
Department of Computer Science
Stanford University
erzeng@stanford.edu

Abstract

Plant diseases present a major threat to agriculture and food security. Modern computer vision techniques have led to the use of convolutional neural networks for plant disease identification as a way to combat this issue. We implement three different CNN models to detect plant disease among common crops given images of their leaves and compare the performance of a custom CNN architecture to that of an AlexNet model trained from scratch and an model AlexNet trained using transfer learning. After training and evaluating each method using F1 score and classification accuracy on a standard PlantVillage dataset, transfer learning was identified as the most effective method of obtaining an accurate classifier with an F1 score of 98.32 and a test accuracy of 98.17. We also conclude that learning from scratch performs very poorly in comparison to the two other models.

1 Introduction

Crop pathogens threaten agricultural production and food security in developing areas of the world. An estimated 22% of wheat, 30% of rice, and 23% of corn yields are lost to pathogens and pests every year [6]. In addition, plant diseases can increase input costs and lead to unnecessary pesticide use that leads to resistant pathogen strains as well as environmental damage through runoff and human health impacts through our food. Identification of crop diseases is a challenge in many areas of the world because of a lack of infrastructure and as a result proper treatment of crops is much more difficult. Using advances in computer vision, a method to diagnose plant diseases prematurely can significantly improve crop yield. Smartphones offer a novel and accessible way for farmers to easily identify these diseases due to their computing power and advanced cameras.

Because data specific to this computer vision problem is sparse relative to the massive domain of general image classification, transfer learning is an appropriate approach to train a plant disease classifier. One candidate we have identified for use is AlexNet, an influential CNN which was originally trained on the ImageNet dataset. Given an input image of a food crop, the classifier will predict whether the plant is healthy, and if not, predict the type of disease. We will build three different classifiers to solve this problem. One will use a custom CNN architecture and will serve as our baseline model. Another will use the AlexNet architecture and be trained from scratch. The last will use transfer learning applied to the AlexNet architecture.

2 Related work

Because of the scale of the problem and open access to large amounts of relevant data, numerous successful implementations of this problem exist. [1] provides a broad overview of methods by

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

describing several approaches using different parameters such as dataset type, choice of training mechanism, and training-testing distribution and lists the accuracy of the resulting combinations. [2] presents a history of previous approaches to the problem and an overview of the different existing deep learning frameworks, their advantages and drawbacks, and their application to plant disease detection. It also notes future work that is needed to improve the limitations of classifiers in real environments and lighting conditions. [3] provides additional evaluation of various CNN architectures and states that current state-of-the-art solutions use deeper networks to achieve a classification accuracy of greater than 99%. [5] gives strategies for training, regularization, and image augmentation that have been shown to help with common problems such as overfitting. [7] reviews the use of hyperspectral imaging, a new technique which can be used to identify onset of plant disease long before classifiers trained on traditional images. However, the lack of public data for hyperspectral leaf images makes this strategy fairly difficult to implement. PlantVillage has also hosted a competition on crowdAI for public submission of various attempts to solve this same problem, many of which achieved >90% accuracy. The ability to make further progress depends on gathering more comprehensive data, as well as customizing deep networks to be suited for this task. Additionally, achieving similar high classification accuracies using more lightweight CNN architectures could prove useful in realistic scenarios through smartphone-based approaches.

3 Dataset and Features

PlantVillage is an open access dataset of plant images started by Penn State University used to monitor crop health. This is the most comprehensive collection of images on plant health on the web. Started in 2013, this dataset now contains over 54,000 samples of healthy and unhealthy plant leaf images. Each of the 38 class labels corresponds to a plant-health status pair. For example, an image of an apple leaf may be labeled as Apple_Black_rot if the leaf has black rot or Apple_healthy if the apple leaf is healthy.

We obtained a preprocessed dataset from GitHub [3]. This is the same PlantVillage dataset used in [1]. Each photo has been cropped to be 256 x 256 pixels with the leaf being the primary object, and processed in 3 different ways. These consist of the raw, color image, a grayscale version, and with object segmentation clearly displaying the leaf on a black background.

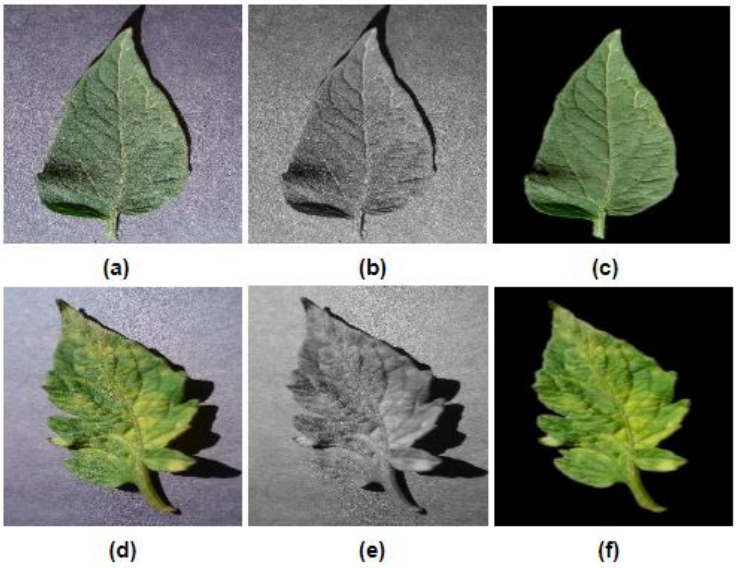

Figure 1: (a-c): Healthy tomato leaf depicted in color, grayscale, and segmented (d-f): tomato leaf with mold depicted in same schemes

The classes categorize 12 different types of plant leaves. Each plant species has different subcategories of diseases. For example, the most prevalent plant: tomato, has 9 different disease classes and 1 for healthy. Indeed, tomato is the most well represented species with about 30% of the examples being tomatoes and 8.7% of those being healthy. Most other plants in the dataset are also fairly well-

represented. However, there are some exceptions with few subcategories. For example, blueberries only have healthy leaf images, and oranges only have one disease class with no healthy images. This mismatch should not be an issue during training and testing. However, in practice, the lack of fully comprehensive classes will impact use cases, if for example, a healthy orange leaf is presented to the classifier. Overall, the number of diseased examples far outnumbers the number of healthy examples. This is a desirable property if we want to achieve a high recall rate.

4 Methods

The baseline model was a CNN with a 3x(3x(Conv-BatchNorm)-Dropout))-Flatten-Dense-BatchNorm-Dropout-Dense architecture. The output layer used softMax classification, as the different plant species-disease pairings are mutually exclusive. We repeatedly stacked convolution layers with a small filter size of 3x3 to both obtain large receptive field views of the input volume and to introduce additional non-linearities for more expressive feature detection. Batch normalization was introduced after every convolution layer to reduce overfitting which was found to be a large issue for the classifiers. Dropout was also added to further reduce overfitting and make the model more robust. One choice of note is the use of a stride of 2 in the third, sixth, and ninth convolution layers instead of max pooling. This similarly downsamples the input but is learnable and takes less computation time than a max pool layer would. An 80%/20% split was used for training and testing. Preprocessing was done by scaling the input data from [0, 255] to [0, 1] and additionally we used rotations, cropping, and shearing to augment the dataset. The model was trained with Adam optimization i.e.
$m = \beta_1 * m + (1 - \beta_1) * dx$
$v = \beta_2 * v + (1 - \beta_2) * (dx^2)$
$x+ = -learning\_rate * m/(\sqrt{v} + \epsilon)$.

The AlexNet model trained from scratch was identical to the custom CNN in preprocessing, except instead using the AlexNet architecture. The transfer learning AlexNet classifier used the AlexNet architecture with ImageNet weights loaded into the model. In addition, stochastic gradient descent with momentum i.e.
$v = \mu * v - learning\_rate * dx$
$x+ = v$
was used instead of Adam. The model was run three times, each with a different number of layers frozen. One had the first two convolution blocks (first 6 layers) frozen. Another had all the convolution blocks (first 12 layers) frozen. The last had everything but the output layer frozen (first 22 layers).

5 Results and Discussion

Below are shown the hyperparameters for the models.

Model	Batch Size	Optimizer	Learning Rate	Decay	Epochs	Dropout	Momentum
Baseline	32	Adam	1e-3	None	25	0.4	None
AlexNet Scratch	32	Adam	8e-5	3e-4	25	0.3	None
AlexNet Transfer	128	SGD w/ momentum	1e-3	4e-3	25	0.5	0.9

Table 1: Hyperparameters

Learning rate, decay, and dropout were chosen using grid search for each of the three models, first over a large range and then again over a smaller range. For the transfer learning AlexNet model, SGD w/ momentum was used instead of Adam which was chosen as a default for the other two models. It was found that SGD w/ momentum performed several percentages better at test time, possibly because there are specific settings where Adam may fail to converge to an optimal solution whereas SGD w/ momentum is a better generalized optimization algorithm and often will more reliably converge. Generally the hyperparameters chosen were all very standard.

We evaluated the models based on F1 score and classification accuracy. F1 score is generally considered the more accurate metric because classification accuracy can overachieve in unbalanced datasets like the PlantVillage dataset by having a higher performance for majority classes. However,

F1 scores and classification accuracies closely matched each other in this case. Below are shown the results for the models.

Model	F1 score	Test accuracy	Train accuracy	Layers frozen	Transfer/Scratch
Baseline	92.72	94.00	98.36	None	Scratch
AlexNet	89.91	90.68	95.21	None	Scratch
AlexNet	98.32	98.17	98.76	6	Transfer
AlexNet	96.85	97.91	97.95	12	Transfer
AlexNet	97.46	97.88	98.12	22	Transfer

Table 2: Performances

With the baseline model we obtained a 94% classification accuracy, with the training and validation accuracy and loss graphs shown.

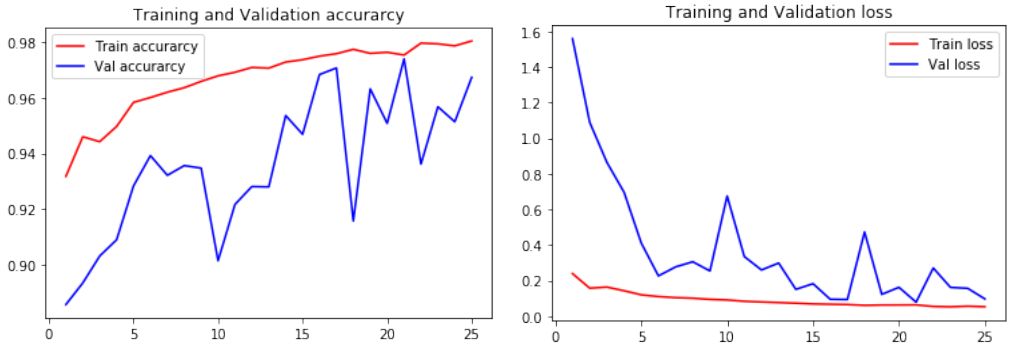

Figure 2: Baseline Accuracy and Loss

Given that the training accuracy reached a much higher 98.3% by the 25th epoch and noting the divergence between the training and validation accuracy curves in Figure 2, it appears that the model overfit the training data. This is odd considering that a fairly large dataset was used. One explanation could be that the baseline model was too complex for the task. Since we started with a high training accuracy and low training loss in the first epoch it could be that a simpler network would be more suited for the classification problem. Stronger regularization and more aggressive dropout could also potentially have reduced overfitting. This classification accuracy seems in line with previous implementations that did not utilize transfer learning, but it does not outperform any model that uses transfer learning from state-of-the-art networks such as ResNet or VGGNet [2] [3].

The AlexNet model trained from scratch only had a 90.68% classification accuracy and performed the worst out of all the models. Although it was not expected to perform worse than the custom model, it isn't that surprising given that the custom model actually has a deeper architecture and more total learnable parameters, as well as similar features such as stacked convolutional layers. In addition, AlexNet was trained on the ImageNet database and while it works well for general image classification, it may not be as well suited for this task.

The transfer learning AlexNet models performed by far the best out of all the models, with the highest performing one having a classification accuracy of 98.17% and the lowest performing one having a classification accuracy of 97.88%. Given that weights were learned on the massive ImageNet database, it is expected that transfer learning would give better results than learning from scratch. Fine-tuning of the model was done by freezing various layers in the network. It is likely that only freezing the first 6 layers was optimal because the first few convolutional layers are very good at detecting generic features such as edges and colored spots because they were trained on the massive and diverse ImageNet database. However, later convolutional layers are more specific to the visual details of plant disease classes and so should be trained on the PlantVillage dataset, especially given that ImageNet data is very different from PlantVillage data. It is expected that for this reason, the models with all convolution blocks frozen and everything except the output layer frozen performed marginally worse as they were less able to detect features specific to plant leave, but were still fairly successful because of the extensive nature of the ImageSet database.

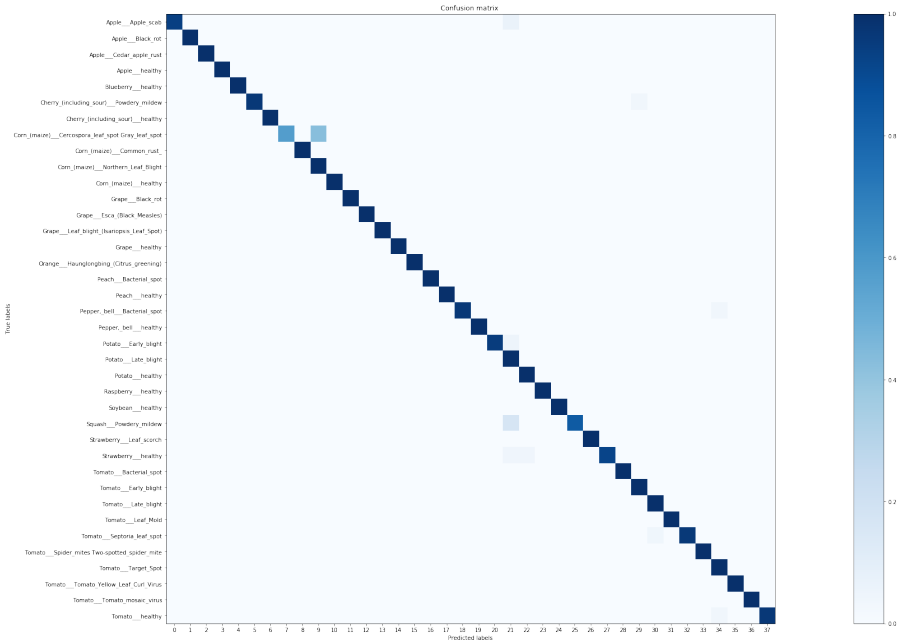

Figure 3: Confusion Matrix of best model

Compared to previous implementations using transfer learning from various architectures, this model performs about the same with accuracy around 98% [1]. While it is less powerful in comparison to the state-of-the-art VGGNet model that exceeds 99% classification accuracy on this task, lightweight networks are of great importance for improving the accessibility of these models towards farmers as smartphone usage is likely to be the future for early plant disease identification [3] [6]. AlexNet has roughly 60 million parameters compared to VGGNet's 140 million, making VGGNet much more computationally expensive, less memory efficient, and a less appealing option for this problem. However, if high performance is the only metric that matters and more computational power is available than what is on a smartphone, VGGNet and other recent CNN architectures will outperform AlexNet.

6 Conclusion/Future Work

Overall the transfer learning AlexNet model with two convolution blocks frozen performed the best in all categories. This was expected given that that the more general earlier layers were trained on the larger ImageNet dataset while the more complex deeper layers were trained on the more relevant PlantVillage dataset. In the future it would be interesting to also use transfer learning with state-of-the-art models such as VGGNet and ResNet and compare the tradeoff between network size and F1 score. Also, if a hyperspectral plant disease dataset becomes available, using that instead of the colored PlantVillage dataset could prove more useful in realistic situations for early detection.

7 Contributions

E.Z. created the baseline architecture, implemented the models, tuned hyperparameters, visualized data, and evalutated results.
C.S. performed past literature review, dataset analysis and preprocessing, and created the video report.

References

[1] Mohanty, Sharada P., et al. “Using Deep Learning for Image-Based Plant Disease Detection.” Frontiers in Plant Science, 22 Sept. 2016, doi:10.3389/fpls.2016.01419.

[2] Saleem, M. H., Potgieter, J., & Mahmood Arif, K. Plant Disease Detection and Classification by Deep Learning. Plants (Basel, Switzerland), 8(11), 468, 2019. https://doi.org/10.3390/plants8110468

[3] Brahimi, M., Arsenovic, M., Laraba, S., Sladojevic, S., Kamel, B., & Moussaoui, A. Deep Learning for Plant Diseases: Detection and Saliency Map Visualisation. 2019. 10.1007/978-3-319-90403-0_6.

[4] Mohanty, Sharada P., “PlantVillage-Dataset”. https://github.com/spMohanty/PlantVillage-Dataset

[5] Mohanty, Sharada P., et al. “Using Deep Learning for Image-Based Plant Disease Detection.” Frontiers in Plant Science, vol. 7, 15 Apr. 2016, doi:10.3389/fpls.2016.01419.

[6] Boulent, Justine, et al. “Convolutional Neural Networks for the Automatic Identification of Plant Diseases.” Frontiers in Plant Science, vol. 10, 23 July 2019, doi:10.3389/fpls.2019.00941.

[7] Kaur, Rajleen, and Sandeep Singh Kang. “An Enhancement in Classifier Support Vector Machine to Improve Plant Disease Detection.” 2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE), 1 Oct. 2015, doi:10.1109/mite.2015.7375303.

[8] Bashish, Dheeb Al, et al. “Detection and Classification of Leaf Diseases Using K-Means-Based Segmentation and Neural-Networks-Based Classification.” Information Technology Journal, vol. 10, no. 2, Feb. 2011, pp. 267–275., doi:10.3923/itj.2011.267.275.

[9] Martín Abadi, et al. "TensorFlow: Large-scale machine learning on heterogeneous systems." 2015. Software available from tensorflow.org.

[10] Chollet, F., et al. Keras. 2015. GitHub. Retrieved from https://github.com/fchollet/keras

[11] Pedregosa et al. Scikit-learn: Machine Learning in Python. JMLR 12, pp. 2825-2830, 2011.