Plant Disease Detection

Sevda SAYAN
Hacettepe University
21527305
b21527305@cs.hacettepe.edu.tr

Fatmanur TURHAN
Hacettepe University
21527429
b21527429@cs.hacettepe.edu.tr

Ismet SEYHAN
Hacettepe University
21693431
b21693431@cs.hacettepe.edu.tr

December 22, 2019

Abstract

In crops produced as a result of agricultural activities, diseases caused by some environmental factors or genetic structure of the plant prevent the productive production. Our main objective in this project is to enable people interested in agriculture to accurately and practically detect diseases in their crops and to implement the most accurate treatment. (1) For this purpose, we have set a target as future work in our project and we will try to make this work a project that can be used on mobile platform.

1 Introduction

Various methods are available to determine if a plant has a disease. One of these methods is to look at the leaves of plants. Discoloration on the leaf, yellowing, fading, desiccation, regional staining, the formation of intermediate holes and such as these symptoms is a sign of disease. Each plant type has its own specific diseases. For example; black rot disease in apple, powdery mildew disease in cherry, common rust disease in corn, leaf blight disease in grapes, etc.

We found two datasets (2) suitable for our project and we combined two datasets and created our own dataset. This resulted in a larger dataset and more classes. In this way, the functionality of our project is increased. The dataset contained a different number of images for each class, and some contained 4000 images. We decided to reduce the dataset to make the transactions easier. It also included photos for the current dataset train and test. So we decided to produce an unseen validation dataset from the test dataset.

Our dataset contains both healthy and diseased plant leaf samples for each plant. There are usually 3 or 4 disease classes for each plant. In total, consists of 38 classes. There are 300 train images, 50 validation images and 50 test images for each class. Our dataset plants: tomato, strawberry, squash, soybean, raspberry, potato, pepper, peach, orange, grape, corn, cherry, blueberry, apple. Figure 1 contains a sample image for each class of our dataset.

When starting the project, articles related to the projects using the same dataset were searched. Related Work section contains information about this article.

2 Related Work

As a result of literature review, articles of similar projects were found. Best article was chosen as references.

According to our reference article(3), 54,306 images of

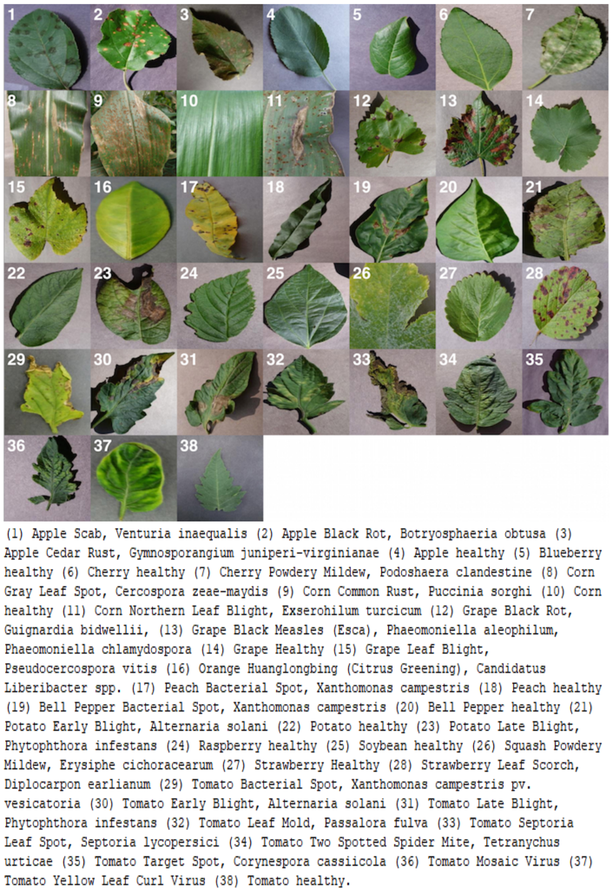

Figure 1: Example of leaf images from the PlantVillage dataset, representing every plant-disease pair used.

plant leaves, which have a spread of 38 class labels assigned to them are analyzed. Each class label is a plant-disease pair and predicts the plant-disease pair from the given image of the plant leaf. In all the approaches described in this paper, the images to 256 x 256 pixels are resized and both the model optimization and predictions on these downscaled images are performed.

Three different versions are used for the whole PlantVillage dataset. Firstly, with the PlantVillage dataset as it is, in color; secondly, experiments with a gray-scaled version of the PlantVillage dataset, and finally, experiments where the leaves were segmented. (Hence removing all the extra background information which might have the potential to introduce some inherent bias in the dataset due to the regularized process of data collection in case of PlantVillage dataset) In Figure 2, sample images from the three different versions of the Plant Village dataset used in various experimental configurations exist.

This set of experiments was intended to comprehend if the neural network really learns the ”concept” of plant diseases, or on the off chance that it is simply learning the inborn inclinations in the dataset.

According to our reference paper, observed that the best results in the classification of plant diseases are obtained by AlexNet and GoogleNet architectures.

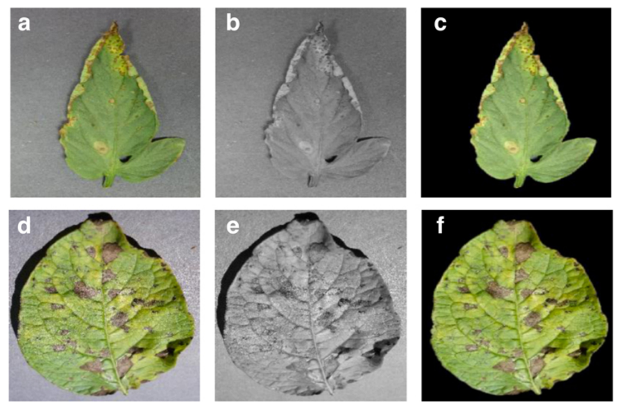

Figure 2: Sample images from the three different versions of the PlantVillage dataset used in various experimental configurations.(A) Leaf 1 color, (B) Leaf 1 grayscale, (C) Leaf 1 segmented, (D) Leaf 2 color, (E) Leaf 2 gray-scale, (F) Leaf 2 segmented.

3 Methodology

We decided to use the usefulness of deep convolutional neural networks for our classification. A convolutional neural network is the most widely used tecnique in the deep neural networks.

Up to this time we focused on VGG16 architecture proposed by K. Simonyan and A. Zisserman from the University of Oxford. They submitted it to ILSVRC-2014 and became one of the famous model. It has some improvements over AlexNet and replaced large kernel-sized filters with multiple 33 kernel-sized filters one after another. In the end it has 2 fully connected layers followed by a softmax. The 16 in its name refers to it has sixteen layers and it is a pretty large network and it has approximately 138 million parameters.

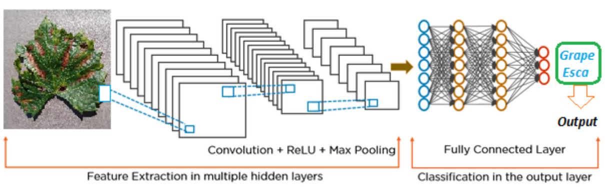

Figure 3: Steps of our network

4 Initial Experimental Settings

We have decided to try some experimental configurations by changing the following parameters:

Architecture

- VGG16
- Resnet50

Training mechanism

- Transfer Learning
- Training from Scratch

As a beginning, we have trained our VGG16 model from scratch by using our belittled dataset. But gotten losses and accuracies was not satisfying. As seen graphs above, train accuracy start with 4% and end with 16% when validation accuracy start with 10% and end with 33%.

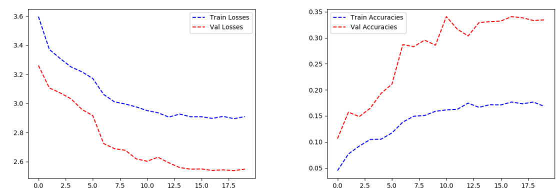

Figure 4: Train and Validation accuracies and losses of VGG16 when training from scratch

Then we have trained VGG16 using transfer learning on our dataset. As a result we obtained higher scores on both train and validation phases. Train accuracy start with 54% and end with 85% when validation accuracy start with 81% and end with 94%.

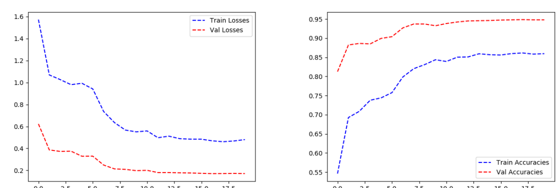

Figure 5: Train and Validation accuracies and losses of VGG16 when transfer learning

In this experiments we used Adam(4) optimizer and learning rate is setted to 0.001. Each experiments runs for a total of 20 epochs, where epoch refers to one cycle through the full training dataset. On each epoch we usually observed decreasing losses and increasing accuracies as it should be.

Among the transfer learning and training from scratch mechanisms, transfer learning consistently performs better than training from scratch and always yields better results which were expected.

5 Future Work

After this point, we will complete our experiments and choose the best model. We are estimating that a model composed of Resnet50 by using transfer learning method will fit to our problem. After that we'll observe class based accuracies and try to improve insufficient ones.

Our next goal is to adapt chosen best model to the mobile platform. We will focus on the correct classification of photos selected from the sd card on a Android device or taken instantly via camera. We will end up integrating our system into the Android environment using PyTorch mobile. PyTorch Mobile, enables developers to deploy any PyTorch model to both Android and iOS.

Deploying A PyTorch model to Android requires the steps below:

- Convert model to TorchScript format (Python)
- Add PyTorch Mobile as a Gradle dependency (Java)
- Load saved model with PyTorch Mobile to perform predictions (Java).

We want to perform this process successfully and get the apk file.

References

[1] J. K. P. R. Kumar, "Advances in image processing for detection of plant diseases." https://drive.google.com/viewerng/viewer?url=https://pdfs.semanticscholar.org/0a10/96ca51b4b8067865211b42c0bec4c9969f34.pdf, June 2011.

[2] "One of the plant disease datasets." https://www.kaggle.com/saroz014/plant-disease.

[3] M. S. Sharada P. Mohanty, David P. Hughes, "Using Deep Learning for Image-Based Plant Disease Detection," 22 September 2016.

[4] "Adam optimizer." https://pytorch.org/docs/stable/_modules/torch/optim/adam.html.