BrainStation Data Science Bootcamp 2022

Final Capstone Report

Casey Nosiglia

4th July 2022

Problem Statement:

- Monitoring plant health can be important in a variety of circumstances. For example, in agriculture, determining the health of a crop (e.g., whether it is healthy or diseased) will have significant influence on the crop yield, and the disease status of the crop will be important information for determining how to proceed (e.g., should the crop be sprayed with pesticide, is it okay as is, etc.). In another example, a gardener may desire to determine if plants in her garden are healthy or not. One easy way to inspect plant health is through a photo that can be taken e.g., with a cellphone, which may then be analyzed to determine the health status of the plant.
- Given this motivation, our problem statement is: ***can we identify, from a photo of the plant's leaves, if a given species of plant is diseased or not with high accuracy, and if it is diseased, can we determine the disease with high accuracy?***
- A smartphone app that would allow a farmer/gardener to identify plant diseases based on a picture of the plant's leaves would thus provide value to the farmer/gardener, giving them more control in taking care of their crop.

Background:

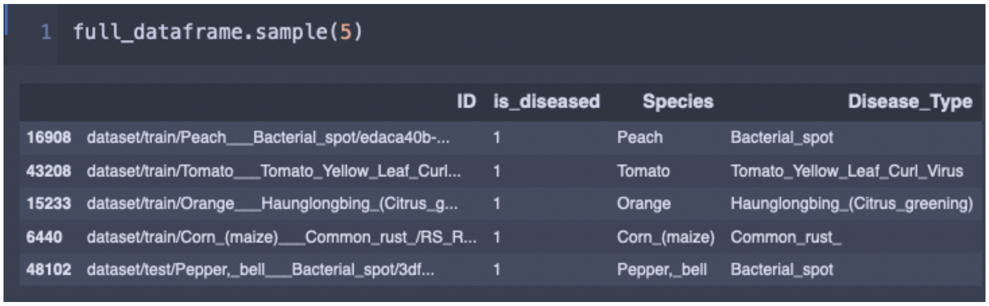
- Image recognition is one of the most celebrated successes in the field of machine learning, and many advances have been made in the field, including in facial recognition, self-driving cars, medical imaging, and many other applications. Thus, this plant disease image classification problem will be more than suitable for data science and machine learning techniques, as it is much simpler than the above-cited examples.
- This problem has been addressed in the past, and in particular, the Plant Disease image dataset (https://www.kaggle.com/datasets/saroz014/plant-disease) is a fairly popular dataset on Kaggle. For example, an earlier version of this dataset (called PlantVillage) has been used to train specialized neural networks in a variety of publications: https://paperswithcode.com/dataset/plantvillage.

Dataset:

- The dataset is the Plant Disease image dataset (linked above).
- The dataset includes ~54.3 256 x 256 pixel images of plant leaves; the total size of the dataset is ~2GB.
- The dataset includes 14 different species of plants, with 38 total categories, which each category being either a healthy category or a diseased category of the 14 species.
- The dataset comes as a file directory, train and test subdirectories, with each subdirectory including 38 subdirectories for each category.

Data Pre-Processing and Analysis:

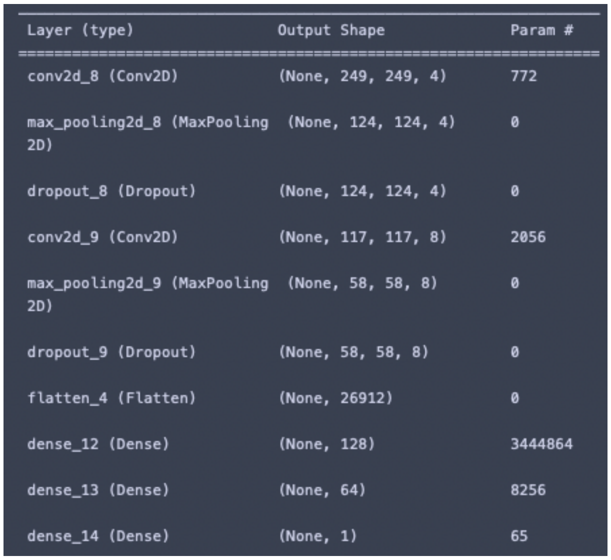
- The dataset is already a well-curated image dataset; however, we needed to create a restructured directory in order to classify images according to (1) diseased/healthy over all species, and (2) healthy/diseased categories for the Tomato species. We created a master 'full_dataframe' in order to manipulate image path data:



```
full_dataframe.sample(5)
```

	ID	is_diseased	Species	Disease_Type
16908	dataset/train/Peach___Bacterial_spot/edaca40b-...	1	Peach	Bacterial_spot
43208	dataset/train/Tomato___Tomato_Yellow_Leaf_Curl...	1	Tomato	Tomato_Yellow_Leaf_Curl_Virus
15233	dataset/train/Orange___Haunglongbing_(Citrus_g...	1	Orange	Haunglongbing_(Citrus_greening)
6440	dataset/train/Corn_(maize)___Common_rust_/RS_R...	1	Corn_(maize)	Common_rust_
48102	dataset/test/Pepper,_bell___Bacterial_spot/3df...	1	Pepper,_bell	Bacterial_spot

- The `full_dataframe` includes the `ID` column (filepath), the `is_diseased` column (1 - diseased, 0 - healthy), the `Species` column, and the `Disease_type` column, and includes 54,305 rows.
- By viewing the frequency of images by 'is_diseased' and by 'Species' and 'Disease_Type', we determined that there were class imbalances, which we corrected with up and down sampling, creating a new `resampled` data frame with 750*38 = 28,500 images. We wanted to remove class imbalances for the classification task, so that the classifier didn't classify based on the relative frequencies of the classes.
- From the 'resampled' data frame, we created resampled (upsampled and downsampled across categories) data frames (in particular) `resampled2`, and `resampled_tomato`. `resampled2` includes 9,000 healthy images (12 categories) and 9,000 diseased images (38-12 = 26), for 18,000 images. There is a slight variance in the counts of the diseased images (ranging from 320-373 per category, where 9,000/26 = 346), as we sampled uniformly from the diseased images; however, each of the healthy image categories have exactly 750.
- 'resampled_tomato' includes 10 categories (9 disease + 1 healthy) with 1,000 images each, for 10,000 images.
- We resampled in order to have even class distribution for modeling; this will not be an issue, as the class frequency does not represent anything significant in this dataset (just the number of pictures taken).
- For the `resampled_tomato` data frame, we created an image directory for the tomato species with 10 categories, with a train/val/test split = 56%/24%/20% => 5,600/2,400/2,000 image split. For the `resampled2` data frame, we created an image directory for healthy/diseased categories, with a train/val/test split = 56%/24%/20% => 10,080/4,320/3,600 image split.



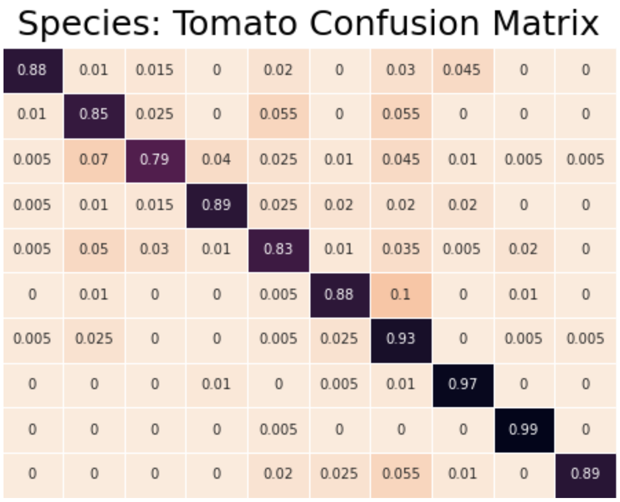
```
Layer (type)                Output Shape              Param #
=================================================================
conv2d_8 (Conv2D)           (None, 249, 249, 4)       772

max_pooling2d_8 (MaxPooling  (None, 124, 124, 4)      0
2D)

dropout_8 (Dropout)         (None, 124, 124, 4)       0

conv2d_9 (Conv2D)           (None, 117, 117, 8)       2056

max_pooling2d_9 (MaxPooling  (None, 58, 58, 8)        0
2D)

dropout_9 (Dropout)         (None, 58, 58, 8)         0

flatten_4 (Flatten)         (None, 26912)             0

dense_12 (Dense)            (None, 128)               3444864

dense_13 (Dense)            (None, 64)                8256

dense_14 (Dense)            (None, 1)                 65
```

Modeling I:

- The first modeling task was to classify images from `resampled2` according to whether they were healthy/diseased. For this task, we created a 6 layer CNN (right), with 2 convolution layers with dropout and max pooling, a flatten layer, and three dense layers (with 128, 64, and 1 nodes, respectively).
- Using the above CNN with data augmentation, batch_size = 32, and 5 epochs, we were able to obtain a train accuracy of 87.99%, with a test accuracy of 89.77%.
- We also generated a confusion matrix, with the diseased label correctly predicted 84% of the time, the healthy label correctly predicted 96% of the time, the healthy label predicted *incorrectly* 16% of the time, and the diseased label predicted incorrectly 4.4% of the time. Thus, in order to improve the model, we would need to be able to identify the more subtle disease images that are confused for healthy images.
- We also tried to model the same CNN except with double the number of convolutions in each convolution layer; however, this proved unsuccessful.

Modeling II:

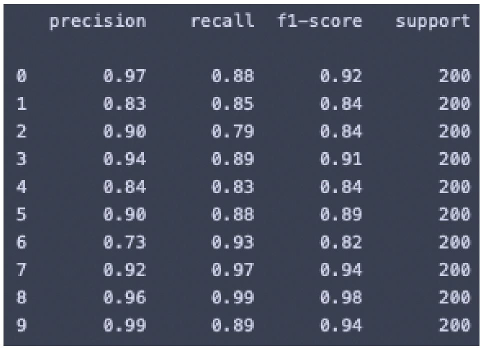
- The second modeling task was to classify images from `resampled_tomato` according to class (1 healthy class with 1000 images, 9 diseased classes with 1000 images each). For this task, we tried two CNNs without much success (see notebook), and then transitioned to transfer learning with ResNet50, as we believed we could get better accuracy with less computation using ResNet50.
- We first tried ResNet50 with frozen layers and a flattening layer, feeding into a 10 node dense layer for classification, with minimal data augmentation. This gave 86.4% test accuracy and 96.7% train accuracy.
- To counteract the above overfitting, we added another dense layer with lots of dropout to the above model, but this led to extremely low accuracy.
- We tried several different rounds of data augmentation. Eventually, we were able to achieve 89% test accuracy and 91% train accuracy over 6 epochs with the original ResNet50 configuration. We then generated the following confusion matrix (right):



Species: Tomato Confusion Matrix

0.88	0.01	0.015	0	0.02	0	0.03	0.045	0	0
0.01	0.85	0.025	0	0.055	0	0.055	0	0	0
0.005	0.07	0.79	0.04	0.025	0.01	0.045	0.01	0.005	0.005
0.005	0.01	0.015	0.89	0.025	0.02	0.02	0.02	0	0
0.005	0.05	0.03	0.01	0.83	0.01	0.035	0.005	0.02	0
0	0.01	0	0	0.005	0.88	0.1	0	0.01	0
0.005	0.025	0	0	0.005	0.025	0.93	0	0.005	0.005
0	0	0	0.01	0	0.005	0.01	0.97	0	0
0	0	0	0	0.005	0	0	0	0.99	0
0	0	0	0	0.02	0.025	0.055	0.01	0	0.89

- The rows of the confusion matrix indicate the true labels, while the columns indicate the predicted labels, with the lowest three accuracies being for classes 2 (Late blight: 79%), 4 (Septoria leaf spot: 83%), and 1 (Early blight: 85%), respectively. For late blight, we note that it was incorrectly identified as early blight 7% of the time, incorrectly identified as target spot 4.5% of the time, and incorrectly identified as leaf mold 4% of the time.
- We also conducted a classification report for the confusion matrix (right):



```
   precision    recall  f1-score   support

0       0.97      0.88      0.92       200
1       0.83      0.85      0.84       200
2       0.90      0.79      0.84       200
3       0.94      0.89      0.91       200
4       0.84      0.83      0.84       200
5       0.90      0.88      0.89       200
6       0.73      0.93      0.82       200
7       0.92      0.97      0.94       200
8       0.96      0.99      0.98       200
9       0.99      0.89      0.94       200
```

- From the classification report, we noted that the lowest three precisions were among the categories (1) class 6 (i.e., target spot) at 73%, (2) class 1 (i.e., early blight) at 83%, and (3) class 4 (i.e., septoria leaf spot) at 84%. Additionally, we noted that the lowest three recalls were among the categories (1) class 2 (i.e., late blight) at 79%, (2) class 4 (i.e., septoria leaf spot) at 83%, and (3) class 1 (i.e., early blight) at 84%.
- This indicates that (1) classes 6, 1, 4 were assigned incorrectly most often, while (2) classes 2, 4, and 1 were missed most often.
- We were able to achieve high accuracy (around 90% for test and train) for both modeling tasks with (relatively) simple models, which is encouraging. Further progress would include more precise identification of subtle spotted patterns (such as for target spot, septoria leaf spot, and spider mites, for example), and also to expand disease classification to other species. For example, the next natural classification task would be to build a classifier for all 38 classes.

Summary:

- By applying CNN and ResNet50 models to augmented and resampled image data of various species of plants, we were able to (1) classify with high accuracy (~90%) whether a leaf was diseased/healthy, and (2) classify with high accuracy (~90%) tomato leaves according to their disease state.
- In order to make this application truly useful, we would have to increase the accuracy to near-perfect (99%+) in order for it to be ready for the market. We would also have to train it on a large variety of plants and diseases for it to be marketable, and allow it to adaptively train according to user input; however, even in our small prototype, we have demonstrated the potential utility of machine learning image classification for monitoring of plant health, which can have practical applications in agriculture.