SPF Report

This report presents two AI models developed for quality check of the ornamental plant Pitósporo. The models were designed to identify the quality of the stems in the field and in bags taken to the warehouse. The goal is to improve the efficiency and accuracy of quality check, reduce waste, and increase the quality of the product delivered to customers. The report will provide details on the development and performance of these two models and on the end, you can find some final considerations.

Field

Data

The objective of the firm in the field is to evaluate, based on a picture, whether a plant can be classified as "Good Leafs Field," "Green Leafs Field," or "Spot Leafs Field." The data that we received consists of a folder with three subfolders, each containing pictures of plants classified by humans as belonging to one of these categories. In total, we have 326 pictures, reasonably distributed across the categories (104, 119, and 103, respectively). Since we are tasked with building a network capable of making these classifications, and assuming we can trust the provided plant classifications, we began by splitting the data.

To train our model, we created a folder called "train," which contains 70% of the data, or 227 pictures. After each epoch, we need to measure how far the model's predictions are from reality and in which direction this distance is minimized. We cannot use the training data for this purpose, as the model has already seen this data, and estimating the metric on training data would overestimate the model's true power. Therefore, a validation dataset is needed for this purpose. We created a folder called "val," which contains 20% of the data or 65 pictures. Finally, after completing the training process, a testing dataset is needed to ensure that the model is not overfitting and to estimate the final performance of the model. Thus, the folder "test" was created with the remaining data, or 36 pictures.

The data in the "train," "val," and "test" folders are also classified by type, so each picture is inserted into the folder of its respective type. It is important to note that the pictures sizes are 2268×4032, so the ratio is 9:16.
Some visual inspection was performed to become more familiar with the data.

Data Augmentation

The firm's aim is to achieve the 90% accuracy on the model's predictions. For the Field's case it turned out to be a challenging target. The amount of data available to train the model is not impressive, so in order to get closer to this target it was decided to apply data augmentation. In this process we apply some predefined transformations to the pictures that we have in order to generate new pictures that can be used to train the model.

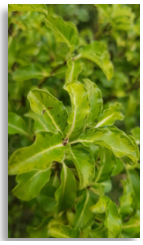

Figure 1 - Original

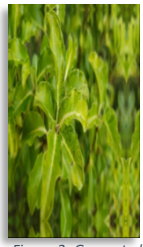

Figure 2- Generated

Model

Our model is based on the VGG16 architecture, which is a convolutional neural network. Since the pictures have a ratio of 9:16, we load the convolutional base with the input shape of (360, 640, 3) to preserve the picture ratio, and the value 3 indicates that we are considering RGB channels. Although the picture quality is high, we have limited computational resources, so we use images with a lower quality to train our model. The CNN results are flattened, and two more fully connected dense layers are added, one with 256 nodes, and the last dense layer has only 3 nodes, each corresponding to one category. The Softmax activation function is used to generate a probability distribution over the classes, with the sum of the probabilities for all categories being equal to 1.

To augment the images with prespecified transformations, the network is trained on image data using an ImageDataGenerator. We load the training and validation sets using the flow_from_directory method from image directories. The weights of the convolutional base layers are set to be non-trainable except for all the layers after and including the 'block5_conv1' layer, this configuration was the most powerful. Therefore, all the layers after and including the 'block5_conv1' layer of the CNN and all the layers on top of it are trainable. We compile the model with the categorical cross-entropy loss function and the Adam optimizer. Since the main part of the CNN is non-trainable, we can use a relatively high learning rate without damaging it. The model is trained for 30 epochs, with model checkpointing callbacks to save the best model based on validation accuracy as 'best_model.h5"'. The following curves were produced.

Results

We can observe from the Loss and Accuracy curves that the training and validation accuracies were close until the 10th epoch, indicating that the model had good generalization. However, on the 18th epoch, there was a drop in the validation accuracy, which was subsequently corrected in the following epoch. Nevertheless, from that point onward, we can observe a widening gap between the training and validation accuracies, suggesting that the model began to overfit, memorizing the training data. The validation accuracy continued to increase, but with a slight drop on the last two epochs. This suggests that instead of using the model trained for all epochs, we should consider the model with the highest validation accuracy. In fact, the test accuracy of the model trained for all epochs was 85.3%, while the accuracy of the "best model" was 91.2%.

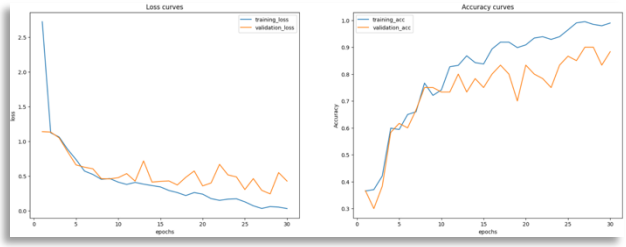

Figure 3 - Field: Loss and Acc. Curves

```
Found 34 images belonging to 3 classes.
2/2 [==============================] - 18s 2s/step - loss: 0.3679 - acc: 0.8529
Test accuracy: 0.8529411554336548
```

Figure 4 - Test Acc. of the model

```
2/2 [==============================] - 18s 2s/step - loss: 0.3385 - acc: 0.9118
Test accuracy: 0.9117646813392639
```

Figure 5 - Test Acc. of the best model

Warehouse

Data

For the Warehouse problem, the provided pictures are classified into only two categories: 'Good Leafs Warehouse – Bag' with 334 pictures and 'Spots Leafs Warehouse – Bag' with 80 pictures. We used the same splitting process as in the previous problem, where 70% of the images were

allocated to the 'train' folder, 20% to the 'val' folder, and the remaining 10% to the 'test' folder. Therefore, the 'train' folder contains 289 pictures, 'val' has 83, and the 'test' folder has the remaining 42 pictures. The classes are imbalanced, which may pose a limitation even though we have only two categories to classify.

Data Augmentation

As with the first problem, we also used data augmentation for the Warehouse problem, applying the same transformations.

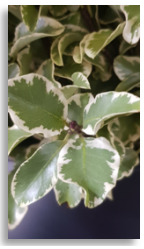

Figure 6 - Original

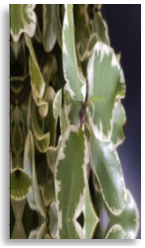

Figure 7 - Generated

Model

Due to the fact that the characteristics that we are capturing are similar, we used exactly the same structure to build the model and trained it for 20 epochs. The following curves were produced.

Results

The curves show that the model achieved a high accuracy quickly, but from the 3rd epoch onwards, there was a noticeable gap between the training and validation accuracies. The validation accuracy plateaued from the 9th epoch, except for the 19th epoch, indicating that the model may have stopped learning any generalizable pattern from the data. Overfitting is suspected, as in the previous case. The test accuracy of the model trained for 20 epochs is 88.1%, but the 'best model' achieved a higher accuracy of 92.86% on the testing data.

Figure 8– Warehouse: Loss and Acc. Curves

```
Found 42 images belonging to 2 classes.
2/2 [==============================] - 20s 5s/step - loss: 0.7645 - acc: 0.8810
Test accuracy: 0.8809523582458496
```

Figure 9 - Test Acc. of the model

```
2/2 [==============================] - 21s 5s/step - loss: 0.1325 - acc: 0.9286
Test accuracy: 0.9285714030265808
```

Figure 10 - Test Acc. of the best model

Final Considerations

Please note that while the models were able to achieve an accuracy of over 90% on the testing data, it's important to keep in mind that the size of the testing datasets was relatively small, so further testing may be necessary to draw more definitive conclusions.

Recommendations

We have observed that deep learning algorithms can be effectively utilized in both the Field and the Warehouse scenarios. However, certain requirements need to be fulfilled to enhance the efficiency of these models.

The primary objective of the firm is to use drones for data collection in the field and for image classification. It is crucial to note that in this case, the model must be trained on data collected under consistent conditions using a drone. Additionally, the quality of the images captured by the drone should be of sufficient quality and taken at a reasonable proximity to the plants. It is also preferable that the images are taken under consistent meteorological conditions and at the same time of day to maintain consistency. In the warehouse, this point is equally crucial, although it may be easier to execute. If it is possible the amount of the pictures should be higher, especially for the case of 'Spots Leafs' which is the category with the lowest number of pictures.

After collecting the data, the plants should be classified by professionals in order to avoid misclassifications, the entire process of model development should be repeated. For the model configuration stage, we recommend utilizing VGG16, which has proven its efficiency in the previous experiments.

To ensure consistent performance, it is essential to monitor the model's accuracy since plant characteristics may change over time, such as in response to climate change. We suggest periodically collecting a subset of the model's classifications and estimating its accuracy. If there is a statistically significant change in accuracy, we recommend retraining the model to maintain or improve its performance.

Note that, depending on the consistency of the data collection between the field and the warehouse, it may be possible to construct a single predictive model that can make accurate predictions for both locations. To achieve this, the input data used for training the model should be collected under similar conditions. However, if this can be accomplished, using all the available data to build a single model has the potential to increase its predictive power even further.

The code can be found on: https://github.com/EugeniuLitvinenco/rep_1