POLITECNICO
MILANO 1863

ARTIFICIAL NEURAL NETWORK AND DEEP LEARNING

Plant Health Assessment with Neural Networks: A Binary Classification Model

Authors:
Naclerio Andrea - 10934883
Pivetta Federico - 10932991
Spolaor Alessandro - 10717380

Professor:
Prof. Boracchi Giacomo

Saturday 18th November, 2023

1 Introduction

1.1 Description of the Classification Problem

In this collaborative effort, our group aimed to devise a deep learning neural network solution for the binary classification of plant health, specifically distinguishing between healthy and unhealthy states. Utilizing high-resolution 96x96 RGB images, the task demanded a nuanced approach to feature extraction and pattern recognition. This undertaking holds particular significance within the domain of precision agriculture, as successful implementation could revolutionize crop monitoring and contribute to proactive disease management. The ensuing paragraphs delve into the technical intricacies of our methodology, highlighting the fusion of deep learning techniques with the specificities of plant health assessment from visual data.

1.2 Description of the Data set

The dataset is composed of 5200 RGB images, each with dimensions of 96x96 pixels. Looking through the different images of the dataset, we have identified 2 outliers; the first image represents Shrek (O1) while the second one is a red/blue image (O2) Since these images are completely not correlated to the binary classification task, we have decided to remove them from the dataset. First, we have tried to discard O2 since the colors were very different with respect to the plant's images. Our idea was to compare the average value of the green channel of the different images with respect to a lower threshold since we would have expected that it would have low green channel values. Contrary to our intuition, not outlier's images were present but only plant's images with dark green color. Since detecting the O1 would have been more difficult since it has very similar colors compared with plants images, we have decided to adopt a more straightforward approach. Scanning the entire dataset, we have identified the position of one image of O1 and one of O2 respectively in the position 58 and 338. So then we have discarded all the images on the dataset that were exactly equal, pixel per pixel, to those. Through this procedure we were able to keep only the plant's images, reducing the dataset to 5004 RGB images, with the same dimension.

Then we analyse the occurrences of the 2 classes in the dataset; we have observed 3101 plant's images classified as healthy and 1903 classified as unhealthy. So the dataset has been balanced, upsampling the unhealthy image through augmentation techniques, using the ImageGenerator function. In particular we have decided as first to split the dataset in a train and a validation set, and apply the augmentation technique only on the training set. If we would have balanced the starting dataset,the splitting would have shuffled the images such that train and validation set it may have had some similar images as result of the augmentation, and so the performance on the validation would present some bias, higher performance, since images in the validation set were very similar to those the model has been trained on.

2 Methodology

While our pursuit aimed at addressing the binary classification challenge of plant health through a deep learning neural network, it is important to note that our approach prioritized the utilization of techniques introduced in our lectures and drawn from relevant scientific literature. While we acknowledge the existence of more advanced methodologies, our methodology focused on a pragmatic integration of fundamental concepts. Emphasizing simplicity and transparency, we exclusively employed Keras models to construct and train our neural network, underscoring our commitment to a comprehensible and accessible implementation within the confines of the knowledge and tools at our disposal.

3 Workflow

3.1 Literature Review

We briefly analyze the main literature results on leaf and plant classification using non-systematic methods. Most of the scientific articles on the use of Deep Learning, DL, in plant classification were trained on the plantVillage dataset which contains images that are quite different from ours (single leaf on a white background). In our study we included the results of [1] and [2]. The first one is a comparative study of DL models for plant disease classification while the other one proposes a CNN to classify leaves.

3.1.1 LeafNet

In [2], Redwan et al. proposed a CNN composed of 5 convolution layers, 5 batch normalization layers, 4 max pooling layers, 1 flatten and 2 dense layers of 800 neurons with dropout equal to 0.5. The proposed network overperformed AlexNet and VGG16 on a novel dataset of mango leaves [2]. We replicated the same architecture even if the original input shape is larger than our (227x227). The smaller input shape results in smaller size of the array after flatten layer. We decided to not reshape the image using bilinear interpolation in order to maintain a lower number of neurons at the end of the convolutional part. Even though we applied different techniques to prevent overfitting (early stopping, low learning rate, reschedule of the learning rate), LeafNet overfitted in all the training attempts. This behaviour is probably due to the large number of trainable parameters (2.38M) and the low number of training images.

3.2 Original-made CNN

We built our own network taking inspiration from LeafNet architecture. To reduce the overall number of trainable parameters we reduced the number of neurons in the dense layers, the size of kernels and the number of channels in the convolutional layers.

Then, we inserted a final dense layer because the network in the first training was underperforming. The number of neurons and channels have been optimized through a grid search approach with random assignment on a reasonable set of variables. Although different technique were implemented to avoid overfitting, the network was not able to classify with the same ability on the train and on the validation set. Moreover, the model's performances was poor with respect to those of pre-trained models.

3.3 Keras bulit-in Models

We tested different built-in models in keras. Initially, the selection has been performed according to [1], then some other models (e.g. EffientNetV2L, MobileNet) have been chosen to try various computational loads. In the following we inserted a brief description of the workflow for the most performing models. Other models (ResNet50, DenseNet169, VGG16, Xception), less performing than these ones, have been tested and their execution and optimization is reported in the notebook.

3.4 Selected Models

3.4.1 MobileNetV2

We have imported the model keeping the pre-trained weights. First, we have tried to attach directly a dense layer with 2 neurons to the latent space in order to predict the output. The performances were not so good and some overfitting was present between training and validation accuracy. So, we have decided to use multi-dense layers, in particular 3 layers with respectively 64,64 and 32 neurons. In this configuration, quite relevant overfitting was present and the performances on the validation set were not able to overcome a 0.81 of accuracy. An additional fine tuning has been tried but the performance has remained the same.

3.4.2 EfficientNetB3

We have imported the model keeping the pretrained weights on ImageNet. As a first try, we have attached an output dense layer with 2 neurons with softmax as activation function, directly to the latent representation provided by the pre-trained model. Moreover, some augmentation was done as preprocess inside the model in addition to the standard preprocess-input of the particular model that we have imported from keras.application. Some overfitting was present and the performances were not so good. One possible problem was related to the fact that applying the same augmentation used for balance the data, they summed up and so the image's configuration was completely different; for this reason we have decided to apply augmentation only to balance the dataset. Consequently, we have tried to add 3 dense layers with respectively 1024, 1024, 512 neurons attached to the latent space of the pre-trained model to help the model in the classification task. In this case, the performance has improved, especially in the training set, but a larger overfitting was present probably due to the difference between the number of parameters; in this configuration, the number of parameters was over 2 million, so the 5000 samples of our dataset are not sufficient to properly tune all the weights. After these considerations, a smaller dense network was attached to the pre-trained model, 3 layers with respectively 64, 64, 32 neurons and a smaller learning rate (0.0001) were used to obtain a smoother decrease of the validation loss reaching an accuracy of 0.84 with some overfitting. Lastly, an additional fine-tuning has been tested; in order to keep the number of parameters low, we unfreeze the last available convolutional layer; the performances have remained the same as in the transfer-learning-only case.

3.4.3 EfficientNetB3-V2

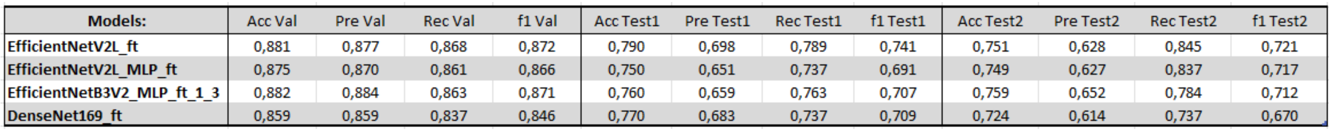
We have also tried another version of the EfficientNet family, EfficientNetB3-V2, a slightly bigger model. We have started with the best configuration of hyperparameter and preprocess used in EfficientNetB3 to see if changing the model we would be able to achieve slightly better performance. Using the model with the pretrained weights and dense network of 3 layers with 64, 64, 32 neurons, the performance on the validation has been increased reaching accuracy that also with the fine tuned in the EfficientNetB3 was not possible to achieve. The best accuracy was 0.86 without any overfitting. Moreover, additional fine tuning has been applied. Due to the small available dataset, we have decided to unfreeze the last available convolutional layer to have a not-so-high number of parameters to be tuned. The performance was dropped completely and looking at the confusion matrix of the final model, the false negative predictions increased a lot. So we have decided to use an unbalanced training set with a higher number of unhealthy plant images trying to improve the performance of the model in correctly classifying the unhealthy plants since it had some difficulties. In particular the number of healthy samples, EfficientNetB3-V2-ft-1-3, has remained the same while the unhealthy one has been triplicate. In this way, the performances have been improved significantly reaching an accuracy of 0.88 and with the lowest number of false negative predictions.

3.4.4 EfficientNetV2L

EfficientNetV2L is one of the largest models available in Keras (119M parameters), we first performed transfer learning and we added a dense layer with 2 neurons and softmax activation for performing binary classification. The accuracy was good and the model was not overfitting. Consequently, to improve the accuracy we performed fine tuning. The number of the layer unlocked has been optimized resulting in good accuracy improvement (from 0.8473 to 0.8809). Nevertheless, the number of false positives (misclassified unhealthy) was higher than false negatives. To improve the correct unhealthy classification we tried to fine tune the model using unbalanced data. However, this solution did not lead to improved results. Parallelly, we tried to improve the performance of the transfer learning model by adding an MLP at the end of the convolutional part of the network. In particular, we added 3 dense layers (128, 64, 32 neurons respectively) with 3 dropout layers (rate=0.3) to prevent overfitting. Although the accuracy of the model was good (0.8681), the model was not as good as the original in classifying unhealthy samples. Therefore, we fine-tuned it using unbalanced data. It improved but it was not able to reach the performance of other models.

4 Results

In figure 1 we reported the results of the models that were performing better on the validation set. Results include the obtained accuracy, precision, recall and f1-score respectively on validation set, test1 set and test2 set. All the models show a reduction in global performance.



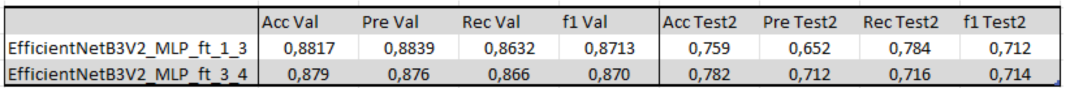
Models:	Acc Val	Pre Val	Rec Val	f1 Val	Acc Test1	Pre Test1	Rec Test1	f1 Test1	Acc Test2	Pre Test2	Rec Test2	f1 Test2
EfficientNetV2L_ft	0,881	0,877	0,868	0,872	0,790	0,698	0,789	0,741	0,751	0,628	0,845	0,721
EfficientNetV2L_MLP_ft	0,875	0,870	0,861	0,866	0,750	0,651	0,737	0,691	0,749	0,627	0,837	0,717
EfficientNetB3V2_MLP_ft_1_3	0,882	0,884	0,863	0,871	0,760	0,659	0,763	0,707	0,759	0,652	0,784	0,712
DenseNet169_ft	0,859	0,859	0,837	0,846	0,770	0,683	0,737	0,709	0,724	0,614	0,737	0,670

Figure 1: Results of the most performing models on the validation set

5 Final Model Selected

Looking at the performances in the test set of figure 1, our best models are EfficientNetB3-V2-ft-1-3 and EfficientNetV2L-ft with an accuracy of 0.759 and 0.751 respectively on the second test. In particular, we can see that precision, in both the models, in the test set has significantly dropped from the validation one, passing from respectively from 0.882 to 0.652 and from 0.877 to 0.628, while recalls remain high. It means that the models are not able to classify correctly the healthy plants thus the number of false positive, healthy plants that are classified as unhealthy, is quite high. For this reason, we have decided to increase both healthy and unhealthy samples; in particular, we set the number of healthy samples 3 times higher while the unhealthy 4 times higher, using the same data augmentation described in par. 1.2, to compensate for the initial unbalancing. The performances on the validation set of the the new model, EfficientNetB3-V2-ft-3-4, are similar to the previous one but those in the test set were better passing from an accuracy of 0.76 to 0.782 but with a precision aligned with recall and f1. The comparison is shown in figure 2.

We could not insert and compare the results of the model EfficientNetV2L-ft since the submission did not return the performances before the delivery of homework. Therefore, EfficientNetB3-V2-ft-3-4 is our final.



	Acc Val	Pre Val	Rec Val	f1 Val	Acc Test2	Pre Test2	Rec Test2	f1 Test2
EfficientNetB3V2_MLP_ft_1_3	0,8817	0,8839	0,8632	0,8713	0,759	0,652	0,784	0,712
EfficientNetB3V2_MLP_ft_3_4	0,879	0,876	0,866	0,870	0,782	0,712	0,716	0,714

Figure 2: Results of the most performing models on the test sets

6 Conclusion

As we can see from the tables in section 4 and 5, despite the implementation of robust techniques such as transfer learning, fine-tuning, and diverse data augmentation methods including rotation and zoom, we can notice a drop in the models' performances between the validation set and the test set. Several factors may contribute to this discrepancy.

One potential reason is the presence of unaccounted variations in the test set that were not adequately represented in the training and validation data. This could include environmental factors, lighting conditions, or unique patterns related to plant health that emerged during the final testing phase. Despite our efforts to develop a comprehensive training strategy, the model may struggle to generalize effectively to these unforeseen variations.

To address this, future improvements could involve expanding the dataset further. A larger and more diverse dataset, encompassing a wider range of environmental conditions and plant health scenarios, could enhance the model's ability to generalize across diverse situations. Additionally, considering ensemble methods, where predictions from multiple models are combined, might provide a more robust and generalized solution to handle unforeseen variations.

Furthermore, exploring advanced preprocessing techniques specific to plant classification, such as disease-specific feature extraction or more sophisticated enhancing methods, could contribute to a more nuanced understanding of the underlying patterns in the images. These improvements, coupled with a more extensive dataset and ensemble methods, could collectively enhance the model's performance and bridge the observed gap between the validation and test sets.

7 Contribution

All the members worked together in the first part of the project. The loading, filtering and management of data phases have been performed altogether and each member through a brainstorming approach contributed to the definition of a workflow. The encoding translation of these phases has also been performed altogether. Regarding the literature search, we approached the problem altogether and the definition of the network has been performed as a group. The same approach has been used to modify LeafNet, obtaining our original-made CNN. Regarding the model definition and optimization, the workload has been split equally among each member. Everybody was in charge of looking at the best strategy to exploit the features of different Keras pre-trained models and optimizing the hyperparameters of the models. More in detail EfficientNetV2L, ResNet50, Xception has been assigned to Pivetta Federico, VGG19, DenseNet169 to Spolaor Alessandro and EfficientNetB3, EfficientNetB3V2, MobileNet to Naclerio Andrea. Finally, the report has been written both separately (each member described his approach to each model assigned) both altogether (par. 1, 2, 3.1, 3.2, 4-7).

References

[1] R. A. Rizvee, T. H. Orpa, A. Ahnaf, *et al.*, "Leafnet: A proficient convolutional neural network for detecting seven prominent mango leaf diseases," *Journal of Agriculture and Food Research*, vol. 14, p. 100 787, 2023, ISSN: 2666-1543. DOI: `https://doi.org/10.1016/j.jafr.2023.100787`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S2666154323002946`.

[2] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," *Computers and Electronics in Agriculture*, vol. 161, pp. 272–279, 2019, BigData and DSS in Agriculture, ISSN: 0168-1699. DOI: `https://doi.org/10.1016/j.compag.2018.03.032`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0168169917313303`.