Artificial Neural Networks and Deep Learning
Homework 1: Image Classification

Ricci Alessandro - 10931743 | Scroccarello Francesco - 10723028
Taccucci Jacopo - 10827846 | Viganò Martina - 10671356

1 Introduction

The challenge addresses the task of image classification in the context of health status of generic plants. Throughout the project, we followed a rigorous testing and verification approach where a set of models was initially chosen based on the proposed classification problem. These models were progressively analyzed and tested using the standard set of performance indices, and then improved on the basis of them. The use of the confusion matrix and the analysis of the performance indices allowed us to understand what our model was weak to and made us think about a specific solution. This approach resulted into a continuous performance improvement, reaching, on the final test, an accuracy of 85,7%, a precision of 79,26%, a F1 score of 81,78% and a recall of 84.47%. Considering the context examined, this last result is of considerable importance: the model can be effectively applied in industrial settings to promptly intervene in plant care.

2 Data preparation

The given dataset consists of 5202 image samples with labels `healthy` and `unhealthy`. An initial analysis brought to our attention the presence of outliers. As a first step, we operated a clean-up by removing these images, bringing the cardinality of the dataset to 5004 samples. In the second step, we addressed the problem of unbalancing. A statistical analysis revealed the persistent disparity between the two sample classes, with up to 1000 more samples in favour of the `healthy` category. To rebalance the dataset, we considered the following solutions:

- Oversampling on the `unhealthy` class: balance at the cost of introducing a bias due to dependency with existing samples.
- Under-sampling on the over-sampled class: balance at the cost of losing some training samples and thus generalisation.
- Unbalanced training: maximum utilisation of the data at the cost of introducing skewness into the model training.

After several tests, we settled for the under-sampled dataset.

3 The model

In the process of building the models, we started from a base model and then improved it using different strategies but maintaining a common structure consisting of the following levels:

- input_layer: layer of shape (96x96x3), so as to handle input images without the needing to perform resizing.

- pre-processing_layer: Sequential structured layer for augmentation purpose with the following functions: `RandomTranslation`, `RandomFlip`, `RandomRotation`,`AutoContrast` and `RandomGaussianBlur`. Their associated parameters have been chosen based on empirical result obtained during the training phase.
- base_model: this is the core of the model, responsible of performing the feature extraction, and is the main layer that was modified in the process.
- output_layer: 2 neurons with softmax as activation function to have a smooth normalization of inferred probabilities.

In addition, we used the technique of EarlyStopping make the training more computationally efficient. All the training process was performed by splitting the dataset into a training set and a validation set, and eventually also a small test set to assess the performance of our models locally. Once we found a model of good quality, it was retrained using the whole dataset in such a way that no samples were lost.

3.1 First Model

For the first attempt we tried to develop our model of a basic Convolutional Neural Network (CNN) using as Feature Extraction Network (FEN) a set of standard volumes:

- 5 convolutional layers
- ReLU activation functions
- 4 max pooling layers and 1 global averaging pooling (GAP) layer in the end instead of a flattening one

We trained it by minimizing the categorical crossentropy function with an Adam optimiser and a learning rate of 10^{-5}. As expected, due to the simplicity of the network, this approach did not work very well, as we achieved an accuracy of 63% on the test set; so, in order to tackle the problem, we resorted into other strategies: transfer learning and fine tuning using the pre-trained models from `keras.applications`, whose weights are trained on `imagenet`.

3.2 MobileNet

The poor results of the first model suggested poor generalization and inefficient use of the dataset. Therefore, we chose to use the transfer learning technique. The choice for the first transfer learning model fell on MobileNetV2 [1]. Its computational lightness coupled with its excellent performance and being based on imagenet makes it an excellent network to be trained on a relatively small dataset and with low domestic computational capabilities. It is also the perfect choice if you want to deploy the model on industrial IoT systems, which is the case of our project. The construction of the model followed the format specified in the previous section, with the use undersampled dataset. The first problem encountered was the heavy overfitting improved with the use of techniques such as Dropout, EarlyStopping and Learning rate scheduling. For further testing we fine tuned the model at different depth levels, but without significant improvement. In total, however, validation accuracy always remained in the 70%, 77% range. This is the main reason why we decided to move to another base_model.

3.3 Keras Applications exploration

We started thinking about other `keras.applications` models and we had other few trials:

- InceptionNet[2]: a very deep network with lots of parameters using the concatenation of parallel filters of different size to reduce the complexity of using a single large filter.
- ResNet[3]: under the impression that the function to learn was very complex and a very deep network couldn't handle that hard problem, we believed that using a network with residual learning and identity mapping simplifies the problem.

Athough our reasoning and research, we kept having poor results.

3.4 ConvNeXt

We had a look at ConvNeXt [4] and we read that it was meant to improve the performance of a standard CNN using also the benefits introduced by ResNet. We decided to have a trial and we reached discrete results, so we maintained this as core model and started working on it. We thought to make an attempt with transfer learning, keeping frozen only the first 60 levels but things did not notably improve, so we tried to re-tune all the layers and we experienced a significant improvement in results. We also started applying batch normalization to the whole training set in order to have more independence between samples, we tried different augmentation functions and strategies; for example we tried avoiding some transformations or using other ones or even we tried applying the whole preprocessing only to the training data not to have that layer within the network and we had different results. In the end, we kept the model with batch normalization applied to the training set and preprocessing layer inside the network. We had pretty good results and on the final testing our best model achieved 85.70% of accuracy.

4 Conclusion

This report described an image classification project in the context of plant health status recognition. Over the course of the term, we examined various model construction and performance optimization solutions. From the results achieved, it is clear that the transfer learning along with fine tuning techniques are certainly a winning choice to have excellent baseline performance despite limited computational capacity. Among the basic models used, ConvNeXt turns out, as we expected, to be much better performing. However, due to the high load demanding, the use of this network may not be feasible in IoT systems, despite its achievements. Contrary case of MobileNet, which has lower performance, but is usable in systems with low computational resources.

5 Contributions

Team Member	Reasoning	Programming	Testing	Total Time
Ricci Alessandro	10h	25h	25h	60h
Scroccarello Francesco	10h	25h	25h	60h
Taccucci Jacopo	10h	25h	25h	60h
Viganò Martina	10h	25h	25h	60h

Table 1: Contributions

References

[1] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications

[2] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna Rethinking the Inception Architecture for Computer Vision

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun Deep Residual Learning for Image Recognition

[4] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, Saining Xie A ConvNet for the 2020s