Artificial Neural Networks and Deep Learning 2021

Homework 1

Prof. Matteo Matteucci

Cristofaro Giulio (Personal Code: 10555172)
Shalby Hazem (Personal Code: 10596243)
Tiu Robert Andrei (Personal Code: 10652209)

POLITECNICO
MILANO 1863

1 Problem statement

The goal of this homework was to classify images of leafs, divided in categories according to the species they belong to. Given the image, the goal is to predict the correct species.

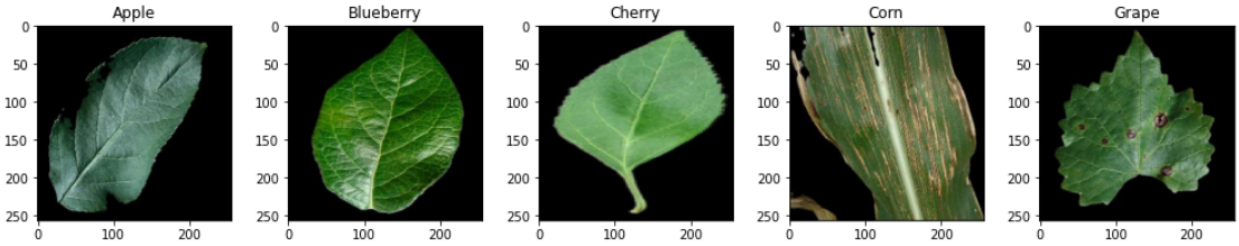


Figure 1: Example of leaves in the dataset

2 The process

2.1 Custom networks without transfer learning

Before trying to use more advanced techniques such as transfer learning and fine tuning, we wanted to understand how far a manually built network could get us.

Given the task at hand, we used the standard architecture of a `Convolutional Neural Network`, consisting of two parts:

- A feature extractor
- A fully connected part for classification

We started off with a very basic network, consisting of `5` sets of convolutional layers in the feature extractor, with a kernel size of `3x3`, stride set as `1x1`, a batch size of `8` and a number of filters starting off at `16`, doubling in each convolutional layer, leading up a maximum of `256` in the last convolutional layer. The dataset was split in three parts (training, validation, test) in order to avoid overfitting.

This was clearly below average, obtaining a mean accuracy of around `23.2%`.

Noticing that the dataset was not extremely big, consisting only of `17728` images, we decided to apply data augmentation. More specifically, we used the `ImageDataGenerator` library options to generate additional rotated, flipped and shifted versions of the original images for the training process. Furthermore, we added some preprocessing to the training images, in the form of rescaling them by a `1/255` factor, in order to make them easier to process by the network. We also increased the batch size in order to obtain better training performance [1].

These precautions led to slightly better results, with a mean accuracy of around `51.3%`.

Although we were getting better results, the accuracy was still quite mediocre, barely beating a coin toss percentage. We decided to tinker with the hyperparameters a bit more, and in an attempt to keep the features more general as possible to avoid overfitting, we removed one convolutional layer, increased the filter size up to a `5x5` mask and reduced the filters number in each layer. We started off with `8` filters in the first convolutional layer and doubled them each new convolution, up to `64` in the last one.

This led to an improved network with a mean accuracy of around `64.1%`.

We tinkered a bit more with the hyperparameters and didn't score any better, so we looked up for different approaches. That's when we decided to introduce a `Global Average Pooling` layer to substitute the `Flattening` layer in the fully connected part of the network. That allowed us to: reduce

overfitting by reducing the trainable parameters (and also speeding up the training process), force the feature maps to be more closely related to the classification categories and at last to make the model more robust to spatial translation in the data.

With these changes, we reached a much more acceptable mean accuracy of `75.8%`.

After some research, we found out that you can achieve increased performance and faster training on some problems by using a learning rate that changes during training [2], so we implemented the Keras callback `ReduceLROnPlateau`, which we set to a patience of `3` epochs and a factor of `0.5` alongside `EarlyStopping` set to a clearly higher patience of `6`. Furthermore, in another attempt to improve generalization, we added weight regularization, using the `L2` vector norm (also called `Weight Decay`), which we set as a `kernel_regularizer` attribute in the `Dense` layer and after a few tests we set the lambda hyper parameter to `0.001`. [3]

This allowed us to improve to our best result of `79.8%`.

2.2 Transfer Learning and Fine Tuning Models

Thinking that we reached the best result that we could with a custom-made model, we looked into the transfer learning and fine tuning techniques to reach even higher scores of mean accuracy.
After some research [4] [5] we concluded that the best known models to use for this kind of task were the `VGG16`, `ResNet50`, `InceptionV3` and the `InceptionResNetV2`.
Various attempts later, the ones that yielded the best results were the `ResNet50` and the `VGG16`, with the latter barely edging out the former in our tests.
Considering that the `GAP` layer improved our performance in the fully connected part of the custom made network, we decided to reintroduce it and, taking inspiration from the actual `VGG16` structure, we connected it with two `Dense` layers consisting of `512` filter each, before the output layer.

This led to a mean accuracy of around `85%`.

To push the network to its limit, we applied fine tuning with a small learning rate (`0.0001`) to it, making the last block of the `VGG16` trainable (freezing `14` layers).
Once again, we paired `EarlyStopping` with the `ReduceLROnPlateau` callback which, as specified, proved very efficient in previous models.

We reached the accuracy peak of `91.3%`, which we considered satisfying.

References

[1] Samuel L. Smith, Pieter-Jan Kindermans, and Chris Ying Quoc V. Le. "Don't decay the learning rate, increase the batch size" (2018). DOI: https://arxiv.org/pdf/1711.00489.pdf.

[2] Abinash Mohanty. "The Subtle Art of Fixing and Modifying Learning Rate" (2019). DOI: https://towardsdatascience.com/the-subtle-art-of-fixing-and-modifying-learning-rate-f1e22b537303.

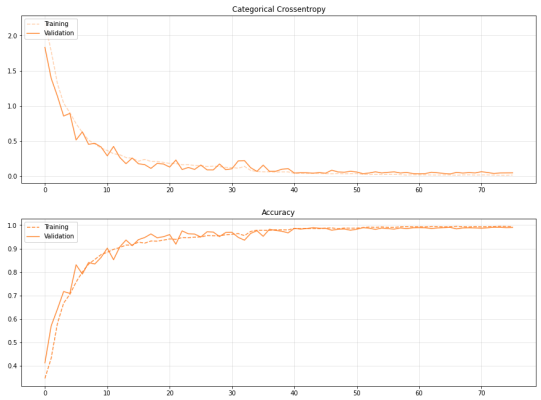
[3] Jason Brownlee. "How to Use Weight Decay to Reduce Overfitting of Neural Network in Keras" (2020). DOI: https://machinelearningmastery.com/how-to-reduce-overfitting-in-deep-learning-with-weight-regularization/.

[4] Jian Gu et al. "Leaf species recognition based on VGG16 networks and transfer learning" (2021). DOI: https://ieeexplore.ieee.org/document/9390789.

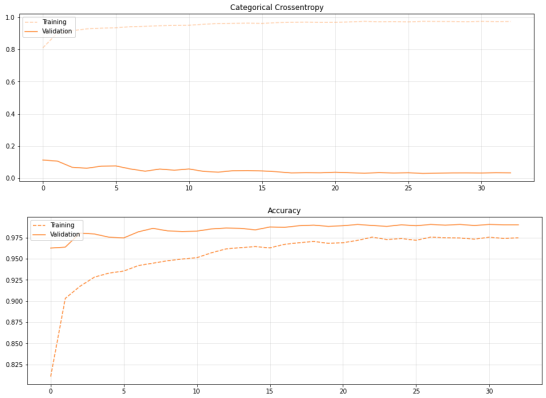
[5] Sk Mahmudul Hassan et al. "Identification of Plant-Leaf Diseases Using CNN and Transfer-Learning Approach" (2021). DOI: https://www.researchgate.net/publication/352264830_Identification_of_Plant-Leaf_Diseases_Using_CNN_and_Transfer-Learning_Approach.

3 Plots

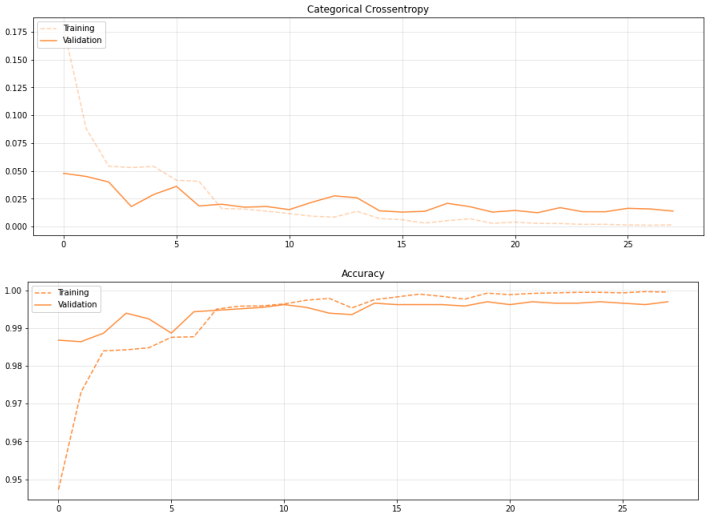
Below are shown the most relevant plots and confusion matrix.



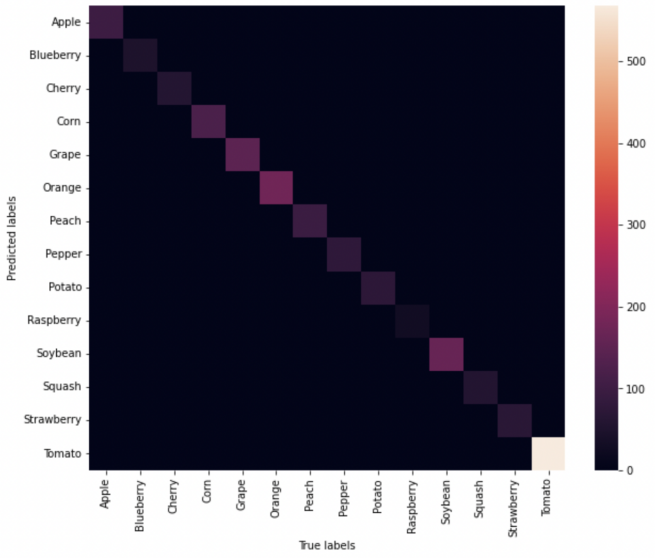
(a) Best custom made model training



(b) `VGG16` Transfer Learning first training



(c) Fine tuning of the previous `VGG16` transfer learning model



(d) Confusion Matrix of the fine tuned `VGG16` model