POLITECNICO
MILANO 1863

Homework 1

Artificial Neural Networks and Deep Learning

Pertusi Federica
Romano Giorgio
Zelioli Chiara

April 2, 2024

1 Introduction

The assignment aims at categorizing plants based on their health condition, involving the creation and implementation of appropriate Convolutional Neural Network architectures. We focused on image augmentation techniques to promote network invariance and improve models accuracy, transfer learning to exploit the robustness of pretrained networks as feature-extractors, and fine tuning to partially adapt those structures to our study. Eventually, we used ensemble and other techniques to improve and merge our models.

2 Dataset description and preprocessing

The dataset constists of 5200 plants images, labeled in two classes, one-hot encoded as healthy = 0 and unhealthy = 1. Images format is RGB, with definition 96x96 pixels. Being RGB's range [0,255], we rescaled to [0,1], to avoid exploding gradients later. The dataset shows some issues such as a small amount of data, which required image augmentation and an unbalanced proportion between classes, approximately 62% healthy to 38% unhealthy subjects, which led us to consider assigning class-weights.
Through visual inspection we identified two types of outliers, which were clearly not images of plants and proceeded to remove them: being able to visualize the whole dataset, we ensured that there were only multiple identical copies of these two extraneous images and saved the index of one copy each. We followed a faster method to compare images than computing pairwise pixel differences by means of the `imagehash` library, hash values were generated for every image and compared to find the copies.
Moreover, 5004 leaves images were kept and splitted in training and validation according to 90% - 10% proportion.

2.1 Data Augmentation and class balancing

The underlying idea of image data augmentation is to artificially train the network on a larger dataset through transformations of training data. We applied random transformations to each image of each batch using `tf.keras.preprocessing.image.ImageDataGenerator` and tuning the hyperparameters range to find the best augmentation parameters according to validation accuracy. Flip, rotation, and zoom turned out to be effective for our case. These transformations highlighted the anomalous features of unhealthy plants, while altering contrast and brightness resulted in an overall

worse images visibility. Making the transformations more aggressive, by increasing hyperparameters range, distorted too much the images and lead to a worse performance of the models.
Furthermore, we carried out a class-specific data augmentation aiming at fixing the imbalance between the two classes, where we had fewer positives (unhealthy subjects) than negatives (healthy). Indeed, it has an impact on the recall ($= TruePositive/(TruePositive + FalseNegative)$) as its relatively low value showed that our models were, initially, not strong at identifying instances of the minority class. Aiming at improving the recall value, we weighed the classes inversely proportionally to the class frequencies through `compute_class_weight` (`class_weight = "balanced"`) imported from `sklearn.utils.class_weight`.

3 Methods and Models

3.1 First attempts

We began with training from scratch a VGG18-like CNN, reaching improvable results in the test (less than 0.7 accuracy) as expected from a simpler model built upon scarcity of data. We then shifted to a MobileNetV2 to make use of transfer learning and fine tuning, on this pre-trained neural network, on ImageNet dataset. However, while it showed promising results during training, we encountered limitations in its performance on the test set. The lightweight architecture of MobileNetV2 seemed to compromise its ability to generalize well during testing. As a result, we opted for heavier models such as ResNet and Inception.

3.2 InceptionV3 and refinements: recall, precision

Moving on, we implemented data augmentation as described above, and explored several Keras Applications models, starting with InceptionV3, a deep CNN that uses small filters to capture local/fine-grained details, and large ones to capture global patterns. To do so, it is structured in modules that combine different filter sizes through multiple parallel convolutions.
We applied fine tuning gradually unfreezing layers in order to improve model performance, without ceding to overfitting, obtaining the so-called InceptionV3_196 model, which turned out to have our best submission accuracy, 0.82. Indeed, enabling the training of more and more layers led to a higher validation accuracy (0.87), which, however, did not correspond to a higher accuracy in the submission.

This last model held the best trade-off between number of trainable layers, in particular from the 196_{th}, and performance on the test set. Moreover, being not satisfied by its recall value, we proceeded to balance the classes as earlier explained: at test time, it grew from 0.68 to 0.71. The precision value, i.e. the proportion of correctly predicted positives out of all the predicted positives($= TruePositives/(TruePositive + FalsePositive)$), deserves a note too. It was 0.8, against an initially low recall, and after the balancing became 0.75, corresponding to an overall inferior accuracy. InceptionV3_196 remained the best model so far.

3.3 InceptionResNetV2 and ConvNeXt

Pursuing the Inception path, we trained InceptionResNetV2, a hybrid network that combines Inception and Residual blocks including shortcut connections to skip layers, designed to mitigate the vanishing gradient problem and easy the training of very deep networks. We also tried a modernization of a standard ResNet toward the design of a Vision Transformer, ConvNeXt (Base and Large). Both models gave quite promising validation accuracy, but did not outperform at test time.

3.4 Final Model

In the second phase of the challenge, the change of test set gave way larger recall than precision for our best model, InceptionV3_196, even reaching 0.8 and 0.6 respectively. We deduced that our model was off-balanced towards the negatives, and we therefore started looking for some techniques to balance our model's performance. In order to improve the precision we tried to use off-balanced class weights to push our model to raise the number of negative predictions. By using class_weights 0.7 and 0.3 we got the InceptionV3_196_70 model. The second attempt to increase the test accuracy was to use an ensemble model. We choose two extremal models, having opposite behaviour, to get a better performing compromise. Specifically, we chose a model having recall and precision values respectively 0.8 and 0.6, and an extreme version of our InceptionV3_196_70 model, trained using class_weights 0.75 and 0.25 and so likely to get a high precision.

4 Conclusions

Despite having tried many different techniques we still did not reach the wished accuracy values. In order to do so we should definitely implement more complex and deeper models, exploiting new techniques as self-supervised learning and mixed precision training, when in need of dealing with model complexity.

5 Contributors

Every member of the group managed to work on models, read documentation and explore techniques while actively participate and discuss to the overall group progress. Therefore, the following indications cannot utterly illustrate the single contributions, as we worked together to interpret each other's results and decide which direction to proceed in.
A rough division could be:

- Federica trained InceptionV3 and ConvNeXt;
- Giorgio worked trasversely, training VGG and MobileNet models and implementing more general techniques like ensemble;
- Chiara trained InceptionResNet, set augmentation hyperparameters and took care of the report.

6 Bibliography

References

[1] Building powerful image classification models using very little data.

[2] Guide to self-supervised learning.

[3] Keras applications.

[4] Kerascv api.

[5] Matteo Matteucci Giacomo Boracchi. Lecture slides. 2023-2024.

[6] Harmandeep Kour and Lal Chand. Healthy and unhealthy leaf classification using convolution neural network and cslbp features. International Journal of Engineering and Advanced Technology, 10:25–31, 10 2020.

[7] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 11966–11976, 2022.

[8] Eugenio Lomurno. AN2DL lectures. 2023-2024.

[9] Santisudha Panigrahi, Anuja Nanda, and Tripti Swarnkar. Deep learning approach for image classification. In 2018 2nd International Conference on Data Science and Business Analytics (ICDSBA), pages 511–516, 2018.

[10] Muhammad Shoaib, Tariq Hussain, Babar Shah, Ihsan Ullah, Sayyed Mudassar Shah, Farman Ali, and Sang Hyun Park. Deep learning-based segmentation and classification of leaf images for detection of tomato plant disease. Frontiers in Plant Science, 13, 2022.

[11] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and ZB Wojna. Rethinking the inception architecture for computer vision. 06 2016.

[3] [4] [2] [5] [8] [1] [7] [9] [11] [6] [10]