POLITECNICO
MILANO 1863

Artificial Neural Networks and Deep Learning
Homework 1 - Image Classification

Frantuma Elia - 10567359 - 945729,
Fucci Tiziano - 10524029 - 946638

A.Y. 2020/2021

Table of contents

Chapter 1

Introduction

1.1 Description of the task

The homework consists in an image segmentation problem on the proposed dataset. In particular, it is required to segment RGB images to distinguish between crop, weeds, and background, as done in the following example.

(1) dataset image

(2) segmented image

The competition is public and organized by the ACRE (Agri-food Competition for Robot Evaluation).

1.2 Dataset

The dataset is composed of images captured by different sensors in different moments and are about two kinds of crops: haricot and maize. Data comes from the 2019 ROSE Challenge where four teams have competed with agricultural robots. Each team has collected images of the same two crops, but in different moments and with different sensors (RGB cameras).

Images in the dataset are divided into different folders based on the team that acquired the image, i.e., Bipbip, Pead, Roseau, Weedelec. For each team, we have two different sub-folders named as the type of crop present in the images, i.e., Haricot and Mais. Finally, for each crop, the captured RGB images (in the Images folder) and the corresponding ground-truth segmentations (in the Masks folder) are provided.

1.2.1 Images

Teams' images share most of the properties but differ from image size and file format.

Shared properties:

- Color space: RGB
- Classes:
  - Crop
  - Weed
  - Background
- Number of Training images (per team per crop): 90
- Number of Test_Dev images (per team per crop): 15
- Number of Test images (per team per crop): 20

1.2.2 Masks

Masks folders contain the ground-truth segmentation for each corresponding (having the same name) image in the Images folder. They have the same exact properties of the Images set apart from the fact that they all have the same file format: PNG.

In each mask, classes are represented by different colors. The dictionary which allows assigning a label to each corresponding color is provided in the starting_kit, as well as the example script in which we show how to transform RGB masks into target masks. In the following, the provided dictionary ('RGBtoTarget.txt' in the starting kit):

- RGB: 0 0 0 - Target 0 (background)
- RGB: 254 124 18 - Target 0 (background)

- RGB: 255 255 255 - Target 1 (crop)
- RGB: 216 67 82 - Target 2 (weed)

A JSON file, containing the labels of the training images, is attached to the dataset. The dataset requires the training images to be divided into three folders (created manually), corresponding to the three target classes. The division was done with a python script, which is contained in the notebooks.

Note: the dataset path expected by the script is:

..\\artificial-neural-networks-and-deep-learning-2020\\MaskDataset

1.2.3 Data augmentation

We have performed data augmentation in order to increase the dataset dimension. Some of the parameters used to perform the transformations are: brightness, zoom, horizontal/vertical shift and flip. We didn't use rotation, since we noticed that the filling could reduce the performance of the classifier.

1.3 Validation set

No automatic validation set is provided. This means that a subset of the training set must be used to perform validation.

In our case, we parametrized the number of training images to be moved into the validation set, with values between 3-15%.

1.4 Test set

The test set was left untouched, apart from the creation of one directory, named "unlabeled", which contains all the images.

1.5 Evaluation

Submissions are evaluated on multiclass accuracy, which is simply the average number of observations with the correct label. To submit a prediction, it is necessary to produce a .csv file containing the predictions associated to each test image. This file has to be submitted to the Kaggle page of the competition to obtain the accuracy score on the test set.

Chapter 2

Neural network architecture

2.1 Ex-novo architecture

First, we tried to build a sequential neural network from scratch: we made many experiments, each one with a slightly different parameter configuration. Among them:

- `start_f`: the number of initial filters;
- `depth`: the number of convolutional layers;
- the learning rate;
- the batch size;
- the size of the input image;
- the shape of the neuron activation function;
- the number of fully connected layers.

After few experiments, we found that the best model was a convolutional neural network having `start_f = 9` and `depth = 7`. The complete model of the network is descripted in the attached file `HomeworkImages.ipynb`.

2.1.1 Score

The best score of the network was 0.85555, obtained with the following settings:

- `start_f = 9`;

- `depth = 7`;
- image resolution: 348x522;
- batch size = 8;
- two fully connected layers, made of 512 and 256 neurons.

2.1.2 Diagrams

Here we show how loss and accuracy changed during the network training, both for the training (grey) and the validation set (orange).

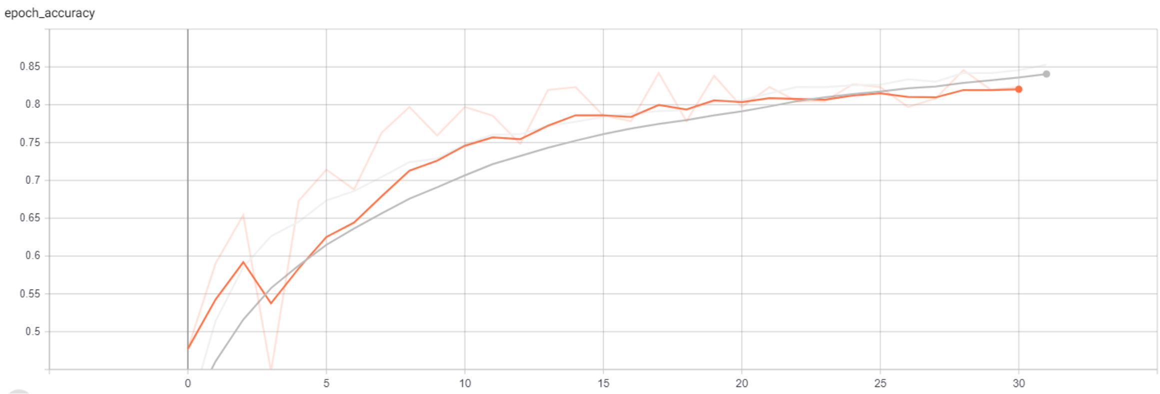

Figure 2.1: Accuracy plot

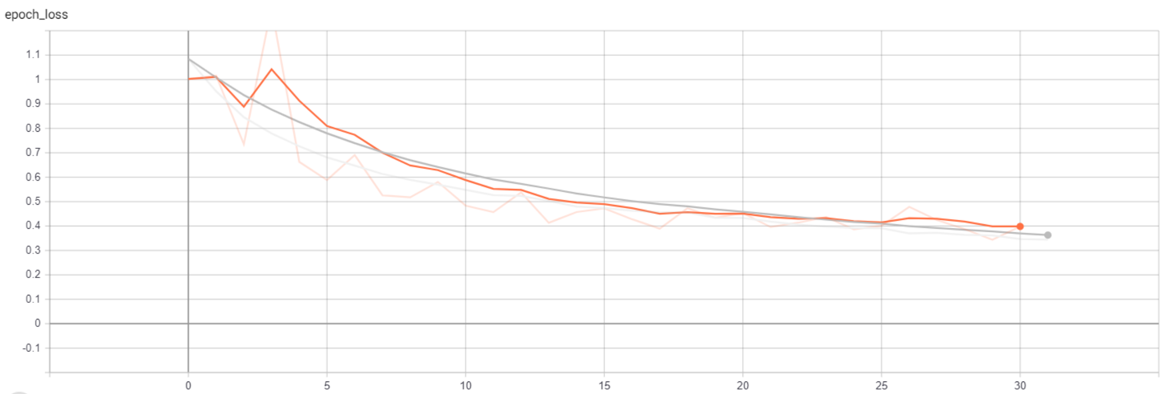

Figure 2.2: Loss plot

2.2 Transfer learning with VGG

The final version of the ex-novo architecture was pretty accurate, but we could not achieve better results by just changing the parameters. We decided to use a pre-trained, reliable and successful architecture.
Our first try was with VGG16, because we've seen it during the lectures. Also in this case we made many experiments, each one with a different parameter configuration:

- `freeze_until`: the layer of the network from which we want to fine-tune;
- the learning rate;
- the size of the input image;
- the shape of the neuron activation function;
- the number of fully connected layers.
- the number of neurons in fully connected layers.

2.2.1 Score

The best score of the network was 0.89333, obtained with the following settings:

- `freeze_until = 11`;
- one fully connnected layer with 32 neurons before the softmax layer.

2.2.2 Diagrams

Here we show how loss and accuracy changed during the network training, both for the training (grey) and the validation set (orange).

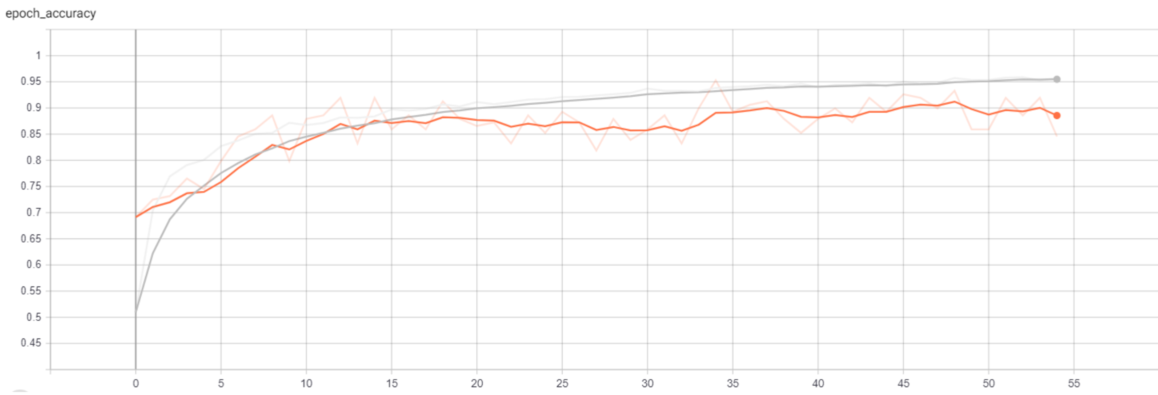

Figure 2.3: Accuracy plot

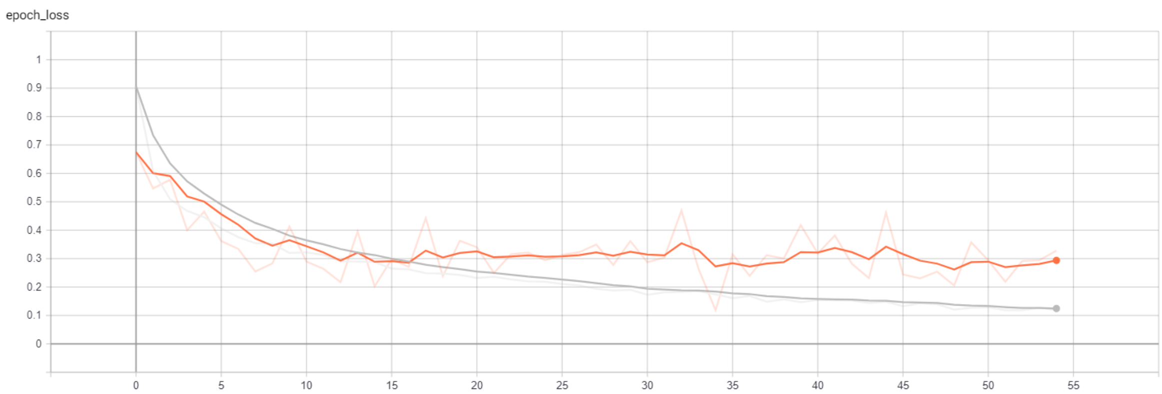

Figure 2.4: Loss plot

2.3 Transfer learning with ImageResNetV2

For our last attempt we decided to use ImageResNetV2, the best performing on the ImageNet dataset. The approach in this case has been similar to the one with VGG, since we only changed the backbone. One significant change that has been made is the replacement of the fully connected layer(s) with a Global Average Pooling layer.

2.3.1 Score

The best score of the network was 0.96888, obtained with the following settings:

- `freeze_until = 150`;

- image resolution: 348x522;
- batch size = 8.

2.3.2 Plots

Here we show how loss and accuracy changed during the network training, both for the training (grey) and the validation set (orange).

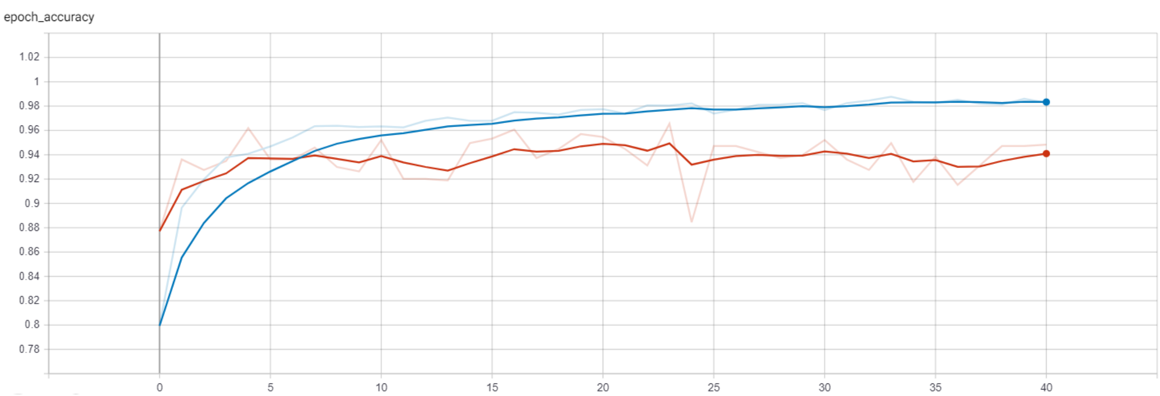

Figure 2.5: Accuracy plot

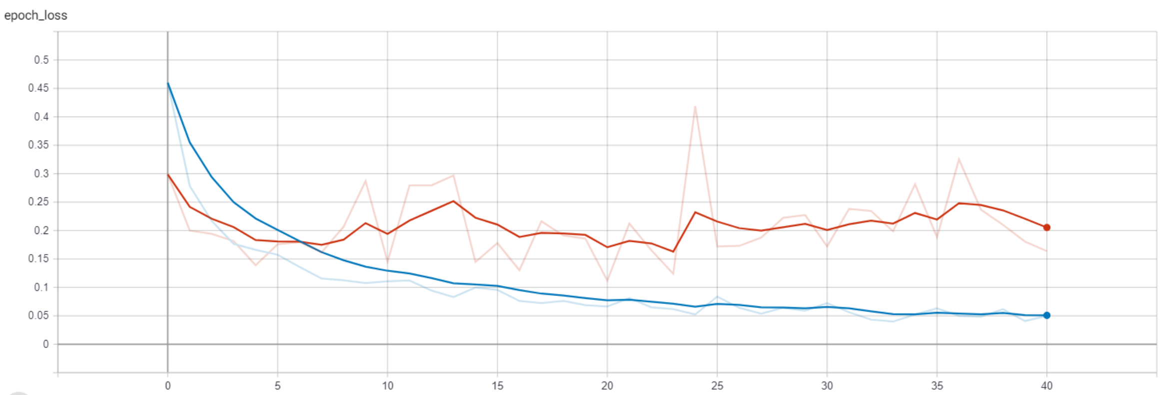

Figure 2.6: Loss plot

Chapter 3

References

3.1 Links

- GitHub repository of the project: `https://github.com/tizianofucci/A2NDLKaggle`