Academic Year 2020/2021

Artificial Neural Networks and Deep Learning

Image Segmentation

Bacchiocchi Francesco 952857
Mameo Edoardo Francesco 944135
Menozzi Fabio 939527

POLITECNICO
MILANO 1863

November 21, 2021

Preprocessing and General Model features

We have decided to focus only on the Bipbip dataset, so we have used only it during training but we have computed predictions also for other datasets obtaining worse results as expected. In all our models we have used a 80%-20% train-validation split with Early Stopping and, sometimes, a learning rate adapter. We have tried data augmentation but didn't improve too much the results of our more complicated models while it increases training time. **We keep separated Haricot and Mais, training one specific model for each of them.**

Basic CNN

We have trained Basic models using images and masks with their original size (2048,1536,3) and allowing the N.N. to have different input sizes (dynamic_input_shape=true) in order to compute the prediction on other datasets. We tried to increase the depth of the model, always checking that the output had same size of the input, but also to increase the number of filters of each block. Since we didn't have too much images, here we decided to apply data augmentation considering also a low batch size. We ended up with a model with 12,583,363 parameters, see notebook **Basic_model.ipynb**

A tiling procedure

When we have decided to go on with more complicated models as a first approach we have still considered for all our models the images with their original size, but due to that we were forced to set the batch-size equal to one to avoid out-of-memory issues, which arised also during predictions. To avoid this kind of problem we have decided to follow a tiling approach, in particular we have cropped every image in 48 subimages having a size of (256,256,3). In a such procedure we have not considered any overlapping and we have finally trained our NN on this new dataset, this also allowed us to increase the batch-size. A scheme of this procedure is presented in figure (1). We have also tried to consider subimages of size (512,512,3) but results didn't improve. Note that also in a prediction phase we have cropped the Test set's images to finally predict and reconstruct their mask. We have done this both for Unet and VGG models. The building procedure of this new datasets can be found in notebook **Tiling_Preprocessing.ipynb**

VGG

Then we have used transfer learning with the VGG16 architecture as encoder, the decoding part was built to obtain a final output with the same dimensions of the input. We have made one trial considering original size images, both from Haricot and Mais together, but the model takes about 4 hours to be trained and so we have decided to go on with tiling to reduce training time. Moreover using input of size (256,256,3) allow us to obtain good results without setting too many VGG-layers as trainable. This model ended up with a total of 16,286,563 parameters of which only 3,931,683 trainable and can be found in **Bipbip_Vgg_Tiling.ipynb**.

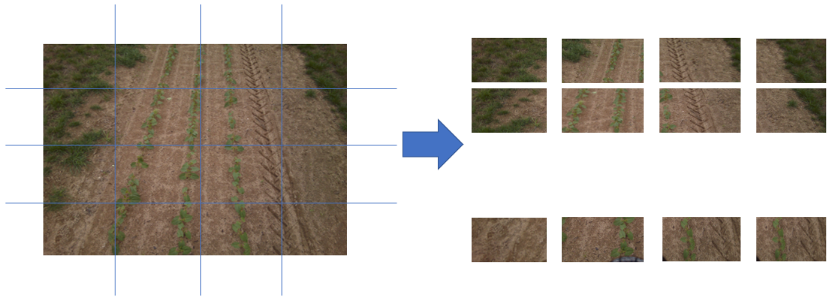

Figure 1: Our tiling procedure to build a new dataset

Unet

In our work we have implemented two different versions of U-net which are slight variations of the standard model.

- In the first model we have considered a number of parameters equal to 7,760,163. With respect to a standard U-net we have added in each block a dropout layer between two consecutive convolution layers considering different dropout rate.
- In the second model we have considered a number of parameters equal to 31,106,595. With respect to the previous one we have added after each Dropout layer a batching layer to avoid overflow problems. We have added one block more both in the downsampling and upsampling part. See notebook **Bipbip_Unet_Tiling.ipynb**.

Table 1: Test Score from Codalab of different models

Model	Bipbip Haricot IoU	Bipbip Mais IoU	Bipbip IoU
Basic model	0.5900	0.7509	0.6865
Vgg original size	0.6488	0.7693	0.7310
Vgg256	0.6171	0.7675	0.7127
Unet256	0.7429	0.8108	0.7894