1st ACRE Cascade Competition Report

Bianchi Chiara, Dominoni Lorenzo, Savino Matteo

For this second challenge we implemented many U-net models for segmentation, different for each specific dataset, to take into account for particular aspects of the data. Here we explain first our base model, and then how we applied it on the various datasets; in the notebooks we left all the implementations we are going to explain, in particular we suggest to look to Weedelec\_Mais files beacuse they contain also the approach from scratch and the weighted loss.

# Base Model

After many trials with FCNN’s we chose that the common aspects of all our models would be:

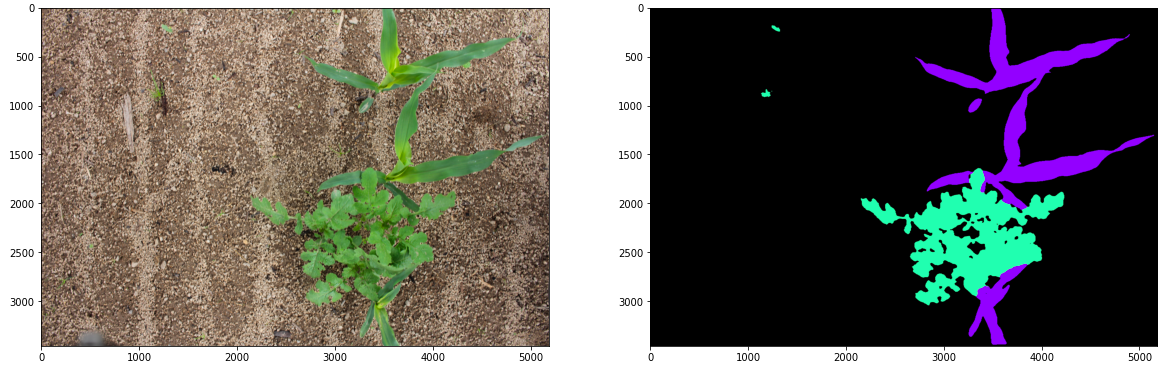
1. ARCHITECTURE: We used a U-net with skip connections focusing on fine layers not to lose spatial resolution, since we think that in our problem the most difficult part is resolution, not semantics. First we implemented a custom architecture trained from scratch (see Weedelec\_Mais) with encoder blocks composed by double convolutions with ReLu, batch normalization and dropout and decoder ones with transpose convolutions, better than standard bilinear upsamplings because they can learn weights. Then we switched to transfer learning, that improved performances using as encoder a B5 efficientnet (near to state of the art and fast enough) with imagenet weights but completely fine-tuned to recover the features useful on this task, and a similar decoder as before, with more dropout to balance the greater complexity of the net.
2. LOSS: We implemented customly the weighted sparse crossentropy (in keras the sparse version is not present), unexpectedly we found worse results, so normal sparse crossentropy was our final choice to optimize the meanIOU.
3. HYPERPARAMETERS: We used early stopping to stop overfitting and decaying learning rate starting from 0.001 to refine the solution, both based on our relevant metric val\_meanIOU.

# Differences Among Models

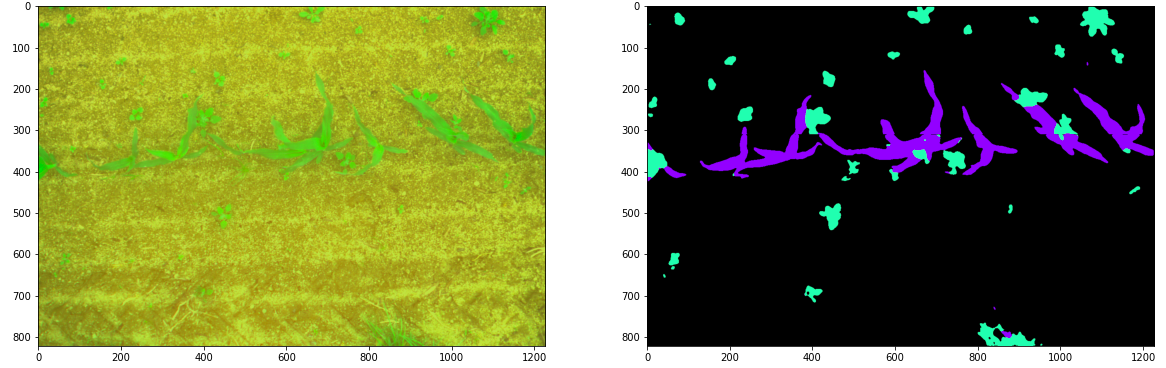
We discovered that the pictures for different plants had similar characteristics, so we trained the same model on Mais and Haricot. In contrast, every team had very different pictures so we had to differentiate models for each one. The main differences are in:

* Resizing or tiling preprocessing
* Data augmentation
* Optimization procedure
* Dropout rate

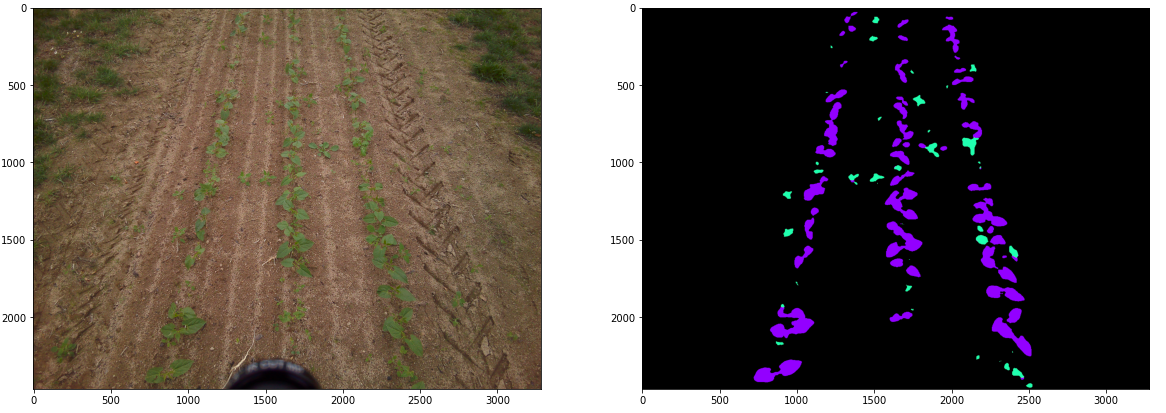
1. WEEDELEC: very big and easy to classify images in which the location of plants is similar and the orientation is the same. So we used resize (otherwise with tiling there were too many only background images) with a size as big as possible to not lose features (1024x1024); nearly no augmentation to not corrupt pictures, and a big dropout rate. In particular we reached 1st place in weedelec\_haricot task with 0.8164 meanIOU



1. ROSEAU: small and bad light images with very different plant locations. We used tiling method with random 512x512 crop at each epoch and bigger batch size to balance for “only background” images. We used also a custom implementation for tiling in test phase. A lot of augmentation was our way to go because of tiling.



1. PEAD: difficult medium size images in which plants location is very similar. Model alike Weedelec with 1024x1024 resizing and only horizontal reflection augmentation (only transformation that made sense). It differed for the optimization technique: we wanted to use Adadelta to converge more precisely, but since it was very slow, we used Adam first and then Adadelta to refine last steps.



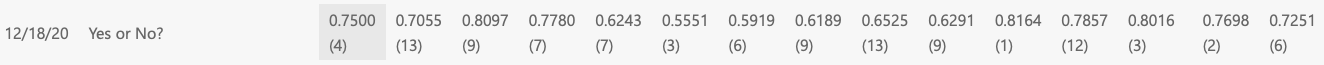
1. BIPBIP: very similar dataset to Weedelec, so we first tried to train those together, but our best results came from different trainings. Same considerations as weedelec apart that here a smaller dropout worked better.

Immagine che contiene schermo

Descrizione generata automaticamente

Thanks to our approach we finished 4th in the overall competition at the end of the second week (20/12) with a meanIOU score of 0.7500, and 2nd in the haricot one with a meanIOU of 0.7698. We think that our successful result is due to:

* Advanced custom U-net architecture with transfer learning
* Deep insight on the images that resulted in specific tricks for distinct datasets



We submitted the same models using all the new data to produce results over the final test set: we obtained 0.7271 meanIOU, a little less than on the previous set, but still good enough, probably because of the hyperparameters tuning ad hoc that we did for the testdev dataset.