Artificial Neural Networks and Deep Learning 2020

Second Competition – Image Segmentation

Team Members: Leonardo Giusti, Francesco Govigli, Lorenzo Mainetti

SCOPE

Given an image dataset, implement a performant algorithm using Neural Networks to segment images into 3 different classes to distinguish between crops, weeds and background.

In particular we must detect:

1) Background, defined as label 0 and has RGB pixel [0, 0, 0] and [254, 124, 18]
2) Crop, defined as label 1 and has RGB pixel [255, 255, 255]
3) Weed, defined as label 2 and has RGB pixel [216, 67, 82]

RGB Image representing Mais crops

Mask Image representing Mais crops

The evaluation is made based on the mean Intersection over Unit (IoU), which is typically used in segmentation tasks and it quantifies the percentage of overlap between the predicted and the target annotation. Results had to be encoded in RLE on a .json file.

MORE DETAILS (Dataset Structure)

The dataset is divided based on the team that took the pictures, i.e. Bipbip, Pead, Roseau, Weedelec. For each team we have 2 subfolders, Haricot and Mais images and the respective masks for segmentation. Each folder is split in training set and test set.

Dataset preprocessing

Initially, to make all reproducible, we inserted a random seed in our notebook. We used a dataframe to encapsule the data tuple (Image.jpg, Image annotation) and split it in a training set of 80% of the original set and in a validation set made of 20% of the samples.

Looking at a first glance to the dataset we notice that there are few pictures in it and in addition the pictures are also of different sizes and with some kind of rotations in the features to analyze. As a result, we decided to make a slight Data Augmentation for the training set varying for instance the rotation_range, zoom_range and horizontal and vertical flip.

Building a baseline

Our first model was built using a pretrained VGG16-Net as encoder with resized images input 256x256 and a decoder of depth = 5 and starting filters = 8, downgrading it by half at each following layer. The prediction layer is simply a convolutional layer with n_filters equal to the number of classes to predict and with a softmax activation function. The baseline reached 0.3 IoU on the test set.

From baseline to our best model performance

Since the upsampling of images in the decoder is a heavy computation which could bring to a higher loss, we decided to reduce the original size of the image to 1024x1024. Moreover, we changed the model, building a custom encoder with 40 starting filters and we decreased the batch size to 1. This was due to the expensive computation of segmentation tasks and higher batches could negatively affect our training computation performances. We trained this model only on the Haricot dataset for BipBip obtaining an initial performance of 0.5 on the test dataset.

Our second attempt was using the U-Net model, but because of the depth complexity and the training data scarcity, we did not obtain satisfying performances.

Eventually, our best model was the one that employed the pre-trained ResNet50 as the encoder, trained only on the Bipbip dataset with an overall performance of 0.659 and a good 0.72 over Mais images. In addition, we noticed that Bipbip and Weedelec have similar images, while other teams have bad quality images and thus, we decided to train our model using both as inputs obtaining Bipbip - 0.6141 and Weedelec - 0.6138 and a total score over the entire dataset of 0.5083. Decreasing the learning rate from 1e-4 to 1e-5 improved slightly the score.

In conclusion, we found out that the batch size was influencing less than the image size for ResNet performances. For this reason, we applied SGD because it is easier to fit into memory, it is computationally fast as only one sample is processed at a time and can help to get out of local minima of the loss function.

Possible improvements

Some possible improvements to our model could be:

- Tiling => to avoid the double resize of the images that contributes to the loss of some information
- Focal Tversky Loss (or other similar weighted losses) => to keep track of the unbalanced task and assign a different weight to crops and weeds (on average more white pixels than red pixels)

Final Submission

For the final submission we trained our best model on the entire dataset with the new images from Test_Dev, now part of the training dataset.

We made some slight changes:

- we now use ResNet152 instead ResNet50.
- we applied the ResNet preprocessing function to our inputs.
- we reduced the Early Stopping patience from 10 to 8.