POLITECNICO
MILANO 1863

Homework 2

Semantic Image Segmentation with Convolutional Neural Networks

Artificial Neural Networks and Deep Learning

Authors

Ali Arslan - CP: *10807090*

Sorrentino Alessandro - CP: *10746269*

Menta Andrea - CP: *10636205*

General approach: Our team concentrated on the *BipBip* dataset, both *mais* and *haricot*. We started by creating a fully customized CNN and moved to transfer learning after. Once obtained satisfying results on both, we moved to a new kind of net called *waterfall* model which bases its main idea on subdividing a multi class segmentation in a set of binary segmentations. Since the dataset was highly unbalanced (in favour of crop wrt. weed), we gave different weights to the two classes in the metrics evaluation. The *mean IoU* metric was weighted in order to obtained a weighted average on IoU of different classes.

Dataset: The dataset has been divided 3 sets: *training*, *validation* and *test* with a 80% - 20% relation. Each class in training and validation roughly contains the same amount of images in order to avoid imbalances. Our python script extracts a certain number of images and saves the names into the two files `val.txt` and `train.txt`. Please place this script inside the BipBip folder.

```
python3 custom_script.py
```

The file `directories.txt` includes all the directories to evaluate all the sub-datasets and produce the final json.

Data augmentation was used in all the experiments with: shifting, rotation, flipping, height and width shift (reflect fill mode), zooming and horizontal flip.

1 Basic custom CNN

Model A (test accuracy: $\approx$ 60% haricot, $\approx$ 72% mais)[1] As custom model, we used a simple encoder/decoder structure with total depth of 10. A single convolutional block is composed by

Encoder:

- Conv2D (3x3, ϕ)
- BatchNormalization
- Relu Activation Function
- Dropout (0.2)
- Conv2D (3x3, ϕ)
- Relu Activation Function
- MaxPooling2D

Decoder:

- UpSampling2D
- Conv2D (3x3, ϕ)
- BatchNormalization
- Relu Activation Function
- Dropout (0.2)

$\Downarrow$

- Output Conv2D (1x1, Softmax)

[1] First model delivered

Where ϕ is the number of filters. Starting from 32 filters in the first block, the number of filters was scaled by a factor of 2 during the up-phase and down-phase. After a few tests we found a good set of hyperparameters: Batch size $\rightarrow$ 2, lr $\rightarrow$ 1e-4, early stopping to mitigate overfitting, reduceLrOnPlateu callback to adjust lr accordingly when near optima, images size $\rightarrow$ 1024x1024.

2 Transfer Learning

Model B (test accuracy: $\approx$ 48% haricot, $\approx$ 63% mais)[2] After reaching good results in the first phase, in which we focused on *custom model*, we decided to try *Transfer Learning*. We started by choosing 2 pre-trained candidate models for the CNN part in order to choose the fittest:

- `Xception`

- `VGG19`

We used the same base configuration for all: *batch size* of 2, same *data augmentation* (except for the *preprocessing function*, proper of each model), fine tuning, same *decoder layers*, *learning rate* of 1e-4, reduceLrOnPlateu and *early stopping* with patience 10 to avoid overfitting.

The decoder part is structured with the following layers:

- Upsampling

- Conv2D (3x3, ϕ)

- BatchNormalization

- ReLU Activation Function

- Dropout (0.2)

- Output Conv2D (1x1, Softmax)

As in the **Model A**, ϕ is the number of filters. Starting from 256 filters in the first block, the number of filters was scaled by a factor of 2 during the up-phase and down-phase.

3 Waterfall Model

Model C (test accuracy: $\approx$ 54% haricot, $\approx$ 78% mais)[3] The basis of this idea is the *"Divide et Impera"* principle. In particular we decided to split our model on two levels:

[2] Second model delivered
[3] Third model delivered

- *Preliminary Level*: with two parallel CNN that perform a *binary segmentation* between background and respectively weed or crop.
- *Merging Level*: one CNN that performs *3-class* segmentation, taking as input an *enriched image*, composed by the concatenation of the *RGB image* and the *2* output masks of the *Preliminary Level*, for a total of 5 layers.

As basis of our CNNs we took into account both *UNet*, *Residual-UNet* and *VGG19*, and after some tests we decided to choose the first one because it gave us better performances. We also tried to apply *Tiling*, but it was not successful, probably because of the unbalancing of the Dataset. At the end we got the following results:

- `Crop-Preliminary Net` 71% IoU
- `Weed-Preliminary Net` 46% IoU
- `Merging Net` 69% mean IoU

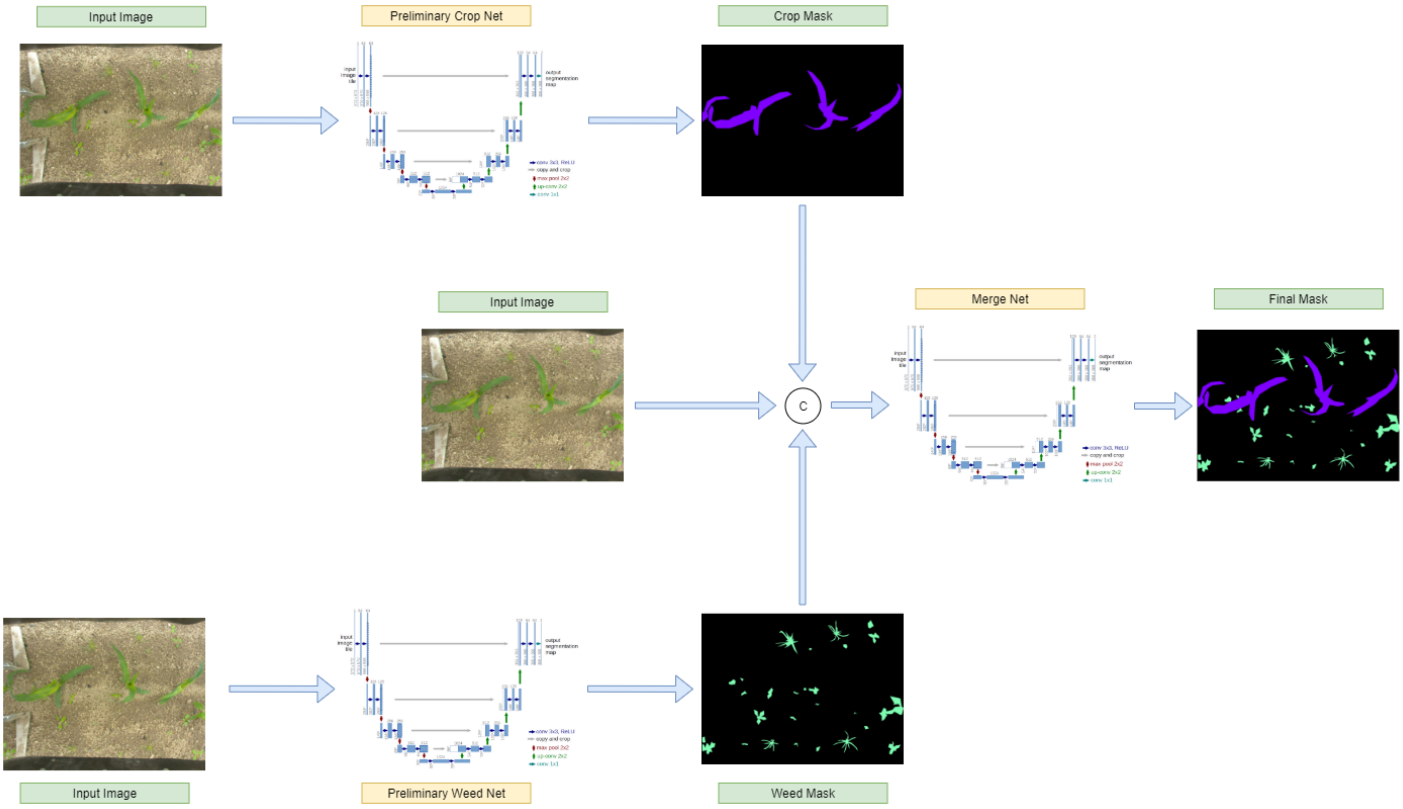

Figure 1: Waterfall model structure

Please, consult the notebooks comments for further specifications.