Artificial Neural Network and Deep Learning
Politecnico di Milano

Project 2 : Image segmentation (long version)

Luca Massini 10504929

Arsène Marzorati 10785084

25-12-2020

1 Introduction

The **ACRE competition** proposes an interesting subject of image segmentation[**course**]. The idea is to use Deep Learning to improve the robot tasks in agriculture. Here, the dataset is provided for two crops (with different tools and different times of the day for the images). The task is to recognize where the crop is (in order to protect it) but also the weed in order to remove it.

We split the work in four parts :

First we wanted to implement a basic network with encoder, decoder and prediction layer and then improve it with some techniques in order to avoid **Overfitting**.

After this basic try, the idea was to use **Transfer Learning** for the encoder part because that part of the network acts like an image classifier and this method was already successfully used in the project for image classification. We used one pre-trained model (VGG16) and used the preprocess for the images.

Then we returned to a basic model, trained on all the datasets, but with skip connections between the encoder and the decoder at each step (implemented as sums). This method allowed us to avoid a too important loss of information with the Max-pooling and Upsampling. We developed 2 models with a slightly different architecture, both trained an all the datasets.

At the end we wanted to mix the improvements with skip connections. For this part we decided to build four networks with the same architecture of the previous one (expect for the number of filters) to be trained on the four different datasets of each project to have four specialized and finally to merge the predictions.

We tried also to build one network for maize and one for haricot and merge the predictions.

2 Treat of the data

We choose two ways for build our training dataset :

- Keep all the images in order to have the largest possible dataset.
- Focus on one project (for the two crops) in order to have specialized network.

We also augmented with data augmentation. We decided to use an aggressive data augmentation process since this kind of task is really rare to obtain a transformed picture that is no more "legal" to be used to train the model.
We had to focus on the meanIoU [4] which is the main criterion for the competition. For that we worked with small batch size (around 10 images) and a learning rate between $1e-4$ and $1e-6$.
We built the dataset class as it was seen in the lab session, a custom data-set class in order to be able to focus on one specific crop or project or to chose to work on all the dataset.
In this costum dataset class, it is possible to choose the validation percentage, the crop to analyze, the project, a possible preprocess for image (useful for Transfer Learning), the size of images and masks and also a generator for reading images and also perform data augmentation.
A similar custom dataset was built in order to read images from the test set (for making the submission).

3 Basic networks for image segmentation

3.1 First Network

A basic network in this kind of task is divided into three parts the encoder, the decoder and the prediction layer. In the first part the network acts as in the image classification [2] and try to class all the pixels between the numbers of classes which are wanted to be detected (here 3). The decoder rebuilt

the image thanks to this classification[1]. This network was really similar to the one seen during the lab lectures.

3.2 Avoid overfitting

In order to avoid overfitting, we used the following methods [3] :

- Dropout has been used in order to avoid that the model generalizes too much its learning. The value used is between 0.2 and 0.4 (part of the data hidden). We have to put in the layer the option seed in order to hid the same part for the mask.
- Early stopping was very important because we chose to use all the data and the training could be very long so with a patience with 5 epochs was necessary.
- Data augmentation has been used in order to have more data available to feed the network. The data augmentation we chose was aggressive in order to have an augmented picture not similar to the original one as much as possible.
- Hyper-parameter tuning: we performed some trials to tune the number of filters of our network in order to choose the right model complexity depending on the data-set size but also on the specific problem.

We tuned the hyper-parameters one by one by doing many trials in which we checked the validation loss depending on the parameter to be changed during the experiments.

3.3 Results

For a basic model, trained with all the data. With batch size of 10, size of input images [512,512], learning rate 1e-5, 5 kind of filters [32,64,128,256,1024]. The plots for the loss, the meanIoU and the accuracy are in the figures 1,2 and 3.
We get only 19.09% with this kind of model. That's why we changed our approach for this task and we moved to transfer learning.

4 Transfer learning for image segmentation

4.1 VGG 16

As we said before, the first part of our model acts like a Classifier for the pixels. In our project of Image Classification we used Transfer Learning with good results. So to improve the previous results, we used the different methods about transfer learning seen in course [3] : the preprocess, function used to transform the image specific for each pre-trained model, fine tuning, only some layers are trainable and we added the other methods to avoid overfitting (Dropout, Early Stopping).
The VGG 16 network was chosen as encoder because it's very easy to build a decoder and so to put methods against overfitting in a efficient way.
With this model we got only a little improvement, 19.36%. The training parameters were :

- Image Size : [512,512] (better than with [256,256]).
- Learning rate : $1e-5$.
- Fine-Tuning : 10O first layers were frozen.
- Batch Size : 10.

The plots for the loss, the meanIoU and the accuracy are in the figures 4,5 and 6.

The two previous kind of networks didn't have any form of skip connections and therefore we didn't have good results. The networks were not able to capture so many details and shapes and sometimes

the predicted pictures were all dark (like if all the pixels were belonging to the background class). So in order to improve this score, we changed our way to work and we moved to a skip connections approach.

5 Skip connections and image segmentation

We started having some interesting results by inserting skip connections in our network. Skip connections have been implemented first as average of the outputs of the encoder and the related part of the decoder and then as sums of them and the second method has been more successful.We started to implement the Upsampling part with transpose convolution with stride equal to (2,2) and the number of filters which reduced of a factor of 2 with the depth. This made the up-sampling process really flexible, powerful and completely learnable by the back-propagation process. The encoding part has been done by increasing the number of filters of a factor of 2 with the depth so to have an equal depth of the decoder and of the encoder.
Then, we had the idea to use different filter sizes both in the encoding part and in the decoding one. We decided this since we wanted the littler filter size convolutional layers to be able to capture little details of the image whereas we wanted the greater ones to be able to capture an overview able to capture rough but useful details. Then the convolutional layers related to different filter sizes were averaged each other so to have a unique volume. We decided to try first by doing a submission using this network architecture trained on all the available datasets. Since the data was a lot and the problem was difficult, using all the datasets, we decided to use a great number of filters and therefore a complex network keeping the same architecture explained above. This single model led us to a results of 0.34 on Codalab.
Then, we tried to improve even more the performances by building 4 specialized networks, each one trained on one of the 4 datasets. Since the data were a lot less with respect to the previous case, we decided to keep the same architecture of before but reducing the number of filters so to have a simpler model that could possibly generalize better. The results were slightly better (0.37).

Then, we tried a different approach in which we built 2 specialized networks, one on Maize and one on Haricot. The network was simpler with just one convolutional layer for each block and therefore a unique filter size of (3,3) and a depth equal to 4 for the encoder and for the decoder. We decided to do so since we observed that one single network for all the datasets was good in some datasets but really bad in others. We merged the predictions of those networks in a unique file and the results were much better (around 42%).

Then we decided to try to train a single network again but with a single kernel size and with 2 convolutional layers concatenated for each block. The network was able to improve the score even better with respect to all the other cases. We achieved a meanIoU value around 0.4862 overall which was our best result in this competition.

6 Conclusion

The task was very interesting and we had time to develop the different methods seen in the course.
The overfitting is a huge problem and if one wants to avoid it and get better results, it's difficult because the training ,in this way and with early stopping, takes more time and memory during the different steps. For example the memory is very important in skip connection and the results are better when the input size of the images is the initial size (no resize) or as similar as possible. But it is impossible on Colaboratory to keep the initial Size.

7 Annex

7.1 Basic Network

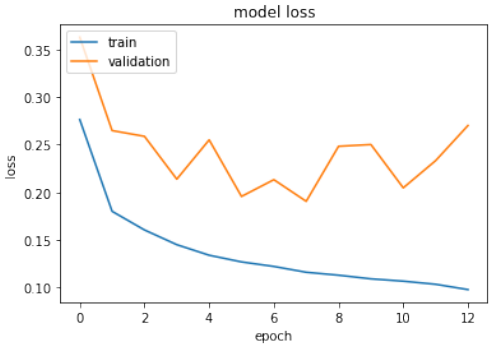

Figure 1: *Loss during the training of basic network*

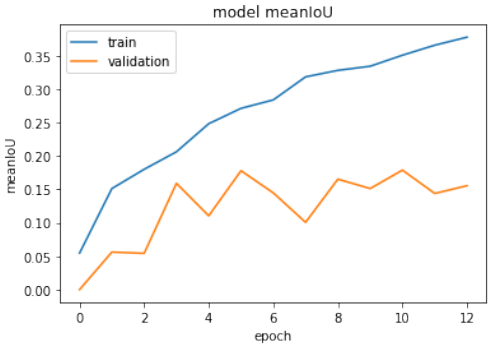

Figure 2: *MeanIoU during the training of basic network*

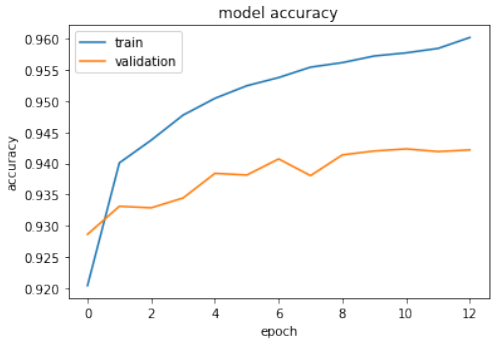

Figure 3: *Accuracy during the training of basic network*

7.2 Transfer Learning

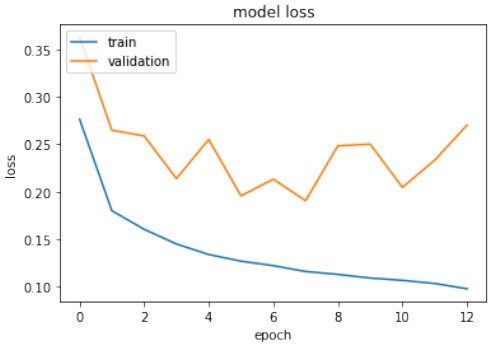

Figure 4: *Loss during the training with Transfer Learning*

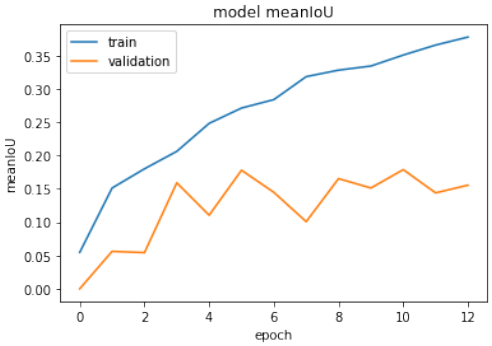

Figure 5: *MeanIoU during the training with Transfer Learning*

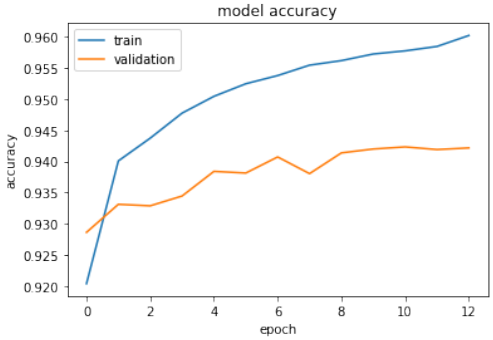

Figure 6: *Accuracy during the training with Transfer Learning*

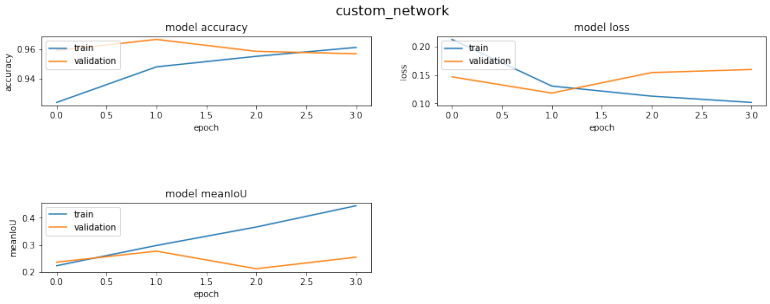

Figure 7: *MeanIoU, accuracy and loss during the training with the first network trained on all datasets (without concatenation)*

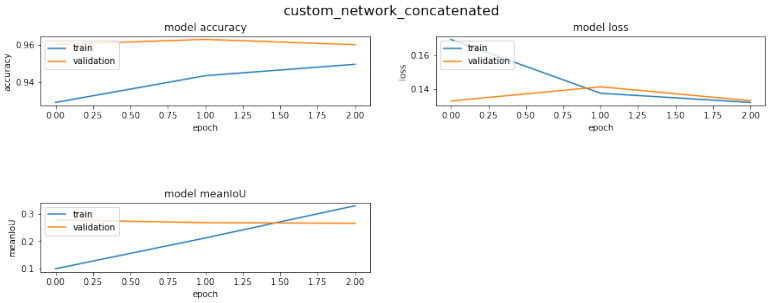

Figure 8: , accuracy and loss during the training with the first network trained on all datasets (with concatenation)

References

[1] Giacomo Boracchi. "Convolutional Neural Networks for Image Segmentation (Course Slides)". 2020.

[2] Giacomo Boracchi. "Convolutional Neural Networks(Course Slides)". 2020.

[3] Giacomo Boracchi. "Training Convolutional Neural Networks(Course Slides)". 2020.

[4] Tiu Ekin. "Metrics to Evaluate your Semantic Segmentation Mode". In: (2019).