AN2DL - Second Homework Report
8pesto

Elisa Nordera, Sara Redaelli, Sveva Zanetti, Rachele Zanin

elisanordera, sarareda17, svevazanetti2, zaninrachele

279637, 275651, 275953, 277461

December 14, 2024

1 Introduction

The aim of the assignment was to develop a deep learning architecture to solve a segmentation problem. More in detail, given 64x128 grayscale images, the task consisted in creating a Neural Network able to categorize pixels correctly, meaning assigning them the proper type of terrain.

To have a better comprehension of the task, our first goal was to **inspect the dataset**. We then focused on **developing from scratch an architecture able to perform segmentation** on images that had never been seen before, and to **evaluate the goodness of the implemented model** we took into account the Mean Intersection over Union metrics on the test set. We started from a standard model and added new features progressively in order to have the network obtain more precise results.

2 Problem Analysis

The dataset we were provided with was made of a training set consisting in 2615 pairs of 64x128 grayscale images from Mars terrain and a mask representing the label of each pixel, and a test set comprising 10022 images of the same type but devoid of their mask.
In particular, pixels could belong to 5 distinct classes according to the type of terrain: ‘Background’, ‘Soil’, ’Bedrock’, ’Sand’ and ’Big Rock’.

As we started to develop our model, we observed that the original dataset needed to undergo a preprocessing phase to get rid of all the outliers that could corrupt the result. After printing some samples of images from the training set we realized that the outliers shared the same label, so we removed them by simply comparing their labels with the standard ones. In this way, we obtained a new training set made of 2505 image-mask couples.
In the cleaned-up dataset we noticed an imbalance between the classes, as label 4 was significantly underrepresented with respect to the other ones, thus we implemented Focal Sparse Categorical Crossentropy to take into account this imbalance.

The most demanding challenge was creating an architecture whose predictions were not too detailed nor too rough, but which resembled the provided annotations. As it is possible to understand in the “**Method**” section, several attempts had to be made to create such a network.

3 Method

After splitting the new dataset into a training set and a validation set, we defined a simple **UNet** only made on two upsampling and two downsampling blocks (both connected one to another by skip-connections) to have a general view of the model's behaviour.

Starting from this input, we tried to develop different models, all based on UNets, to see whether we could improve the obtained results. The first thing we did was to add new upsampling and downsampling blocks in order to deepen the network and make it able to catch more detailed information.
We then came up with the idea of **combining more UNets**, so we implemented a few models with different features, namely two stacked UNets, two parallel UNets (symmetric in one case, with different depths in another one) branching from the same input layer and then concatenating their outputs, and two parallel UNets separately trained.

Moreover, both outside and inside the model, we experimented with different kinds of augmentation, such as brightness, contrast, left-right flip, up-down flip, rotation, and Gaussian noise, but we found out that these techniques did not improve the performance, in fact, they slightly made it worse. Hence, we omitted augmentation in the final model.

However, after a lot of tries involving changes also in the hyperparameters, we came to the conclusion that the best option was **using one single UNet**, as all these architectures were too complex, so, while looking for highly-detailed information, they could not infer labels properly.
Therefore, we concentrated on improving a three-blocks UNet to compromise between finding details and understanding more general features. We observed that adding a combination of **Layer Normalization** and **Batch Normalization** after the convolutional layers had a positive impact on the model performance, as well as adding dropouts at the end of the upsampling blocks. Slight improvements were also achieved through dilated convolutions in the bottleneck, which broadened the receptive field.

As for the loss function, we made some tries using **Focal Loss**, Dice Loss, Sparse Crossentropy Loss or a weighted combination of all of them, but the best results were reached by the first one, which was implemented in our final model. We made different attempts to custom the weights of the five classes too. After realizing that having a really high weight in class 4 negatively influenced the performance, we decided to decrease it in a range between 5 and 20.

3.1 Postprocessing

During the developing phase, we came up with the idea of replacing all labels of class 0 with class 4 (the most underrepresented one) because of two reasons. Firstly, the accuracy of the model is based on the Mean Intersection over Union metrics, which does not take into account the background predictions. Secondly, we realized that the background class was assigned also to labels which could have been better classified. In this way the performance of the model rose from about 0.50 to 0.60.

Nevertheless, after different tries we decided not to change the labels before training. Instead, we introduced a **post-processing function** to modify the predicted ones, achieving even better results: this function took pixels classified as 0 and calculated their Euclidean distance from not-zero pixels and then replaced their label with the one of the closest pixel. As a result, we expanded the area of the pixels belonging to the classes which influenced MIoU, coherently with the predictions of the model. Furthermore, since some labels were composed only by 0 and therefore were not affected by this function, we tried setting them to 4 and noticed a big improvement, reaching a score of 0.68183.

Later on, we thought that an alternative way not to predict label-0 pixels was to set to 0 the weight of the background class and, by doing so, we achieved our best result (0.68543).
We can see in Figure 1 how using different methods affected labels' predictions.

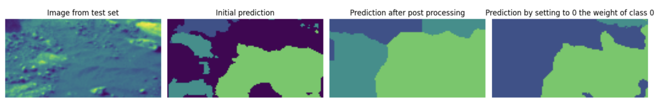
Image from test set
Initial prediction
Prediction after post processing
Prediction by setting to 0 the weight of class 0

Figure 1: Example of the masks predicted through different approaches.

Table 1: Results obtained with different parameters on the network based on a single three-blocks UNet. Submitted models highlighted in **bold**.

Model's features	Scoring
Focal Crossentropy + weight['4'] set to 20 + weight['0'] set to 0	**0.68543**
Focal Crossentropy + weight['4'] set to 20 + weight['0'] set to 0 + augmentation (rotation)	0.67039
Focal Crossentropy + weight['4'] set to 5 + postprocessing	**0.68183**
Combined loss + postprocessing	0.59103

3.2 Hyperparameters

Once we found the most satisfying model, defining suitable parameters for the training phase was quite a challenge. Indeed, we tested several combinations to raise the Mean Intersection over Union.
In the end, based on empirical results, we opted for using batches of 32 samples and kernel size equal to 3. We also decided to use AdamW as optimizer, since it lead to convergence quite fast, and an adaptive learning rate starting from 1e-3.

4 Experiments

The most significant results are presented in Table 1 and Table 2.

Table 2: Results achieved by different architectures.

Model	Scoring
2 stacked UNets	0.61643
2 symmetric parallel UNets	0.49538
2 unbalanced UNets	0.62360
2 separate UNets + postprocessing	0.66331

5 Results

We chose to submit to the final phase two slightly different networks, both scoring above 0.68, one setting to 0 the background's weight during training and the other one applying the post-processing function illustrated in "**Postprocessing**".

6 Discussion

Creating a model from scratch was quite a challenge. Finding the best methods to develop an architecture that could predict the right labels without overfitting involved several tries, most of which were far from successful, and required great effort and creativity.
However, given the complexity of the task, we can say that our final model was discretely good at recognizing different rock types from Mars terrain. There would be a lot of improvements that could help in making predictions more accurate, for instance trying other combinations of hyperparameters, performing other kinds of augmentation to provide the network with other examples to learn from and defining more suitable loss functions.
We also think that having more powerful hardware tools would diminish the amount of time required for the training and thus give us the chance to better explore other options.

7 Conclusions

Our final model was the result of teamwork in which everyone put a lot of effort, searching for literature to find new ideas and contributing to the network's development.
To synthesize the main contribution of each member, we can say that Sveva was in charge of the coding part and came up with the idea of the postprocessing, Elisa was involved in the dataset preprocessing and in adjusting the UNet's parameters, Sara took care of augmentation, training phase and revision, Rachele tested different combinations of multiple UNets and wrote the report.

8 Bibliography

Apart from lecture notes taken during the AN2DL course, we based our hypotheses and our model's implementation on the following articles and papers:

@Computer Science University of Maryland, Stacked U-Nets: a simple network for image seg-

mentation at https://www.cs.umd.edu/~tomg/project/sunets/
@ResearchGate, Enhancing Diabetic Retinopathy Detection with CNN-Based Models: A Comparative Study of UNET and Stacked UNET Architectures at https://www.researchgate.net/publication/385529159_Enhancing_Diabetic_Retinopathy_Detection_with_CNN-Based_Models_A_Comparative_Study_of_UNET_and_Stacked_UNET_Architectures
@Science Direct, Stacked U-Nets with self-assisted priors towards robust correction of rigid motion artifact in brain MRI at https://www.sciencedirect.com/science/article/pii/S1053811922005286