AN2DL - Second Homework Report
OverFritti

Angelo Baturi, Daniele Bergamaschi, Lorenzo Bottelli, Gianluca Carta

1 Introduction

The challenge consists in a semantic segmentation problem, where the goal is to assign the correct class label to each mask pixel. We were provided with a dataset containing 64x128 grayscaled images of Mars terrain where pixels are categorized into five classes, each representing a different type of terrain.

2 Data analysis

The original dataset was composed of 2615 images, and contained some outliers that we decided to remove because they felt a bit "alienated". Then we analyzed the resulting dataset and we investigated meaningful statistics that could help us in designing our solutions. The first thing we inspected was class distribution, and we noticed two important things:

- class 0 was the second-most frequent, but being the background class it contains meaningless information with respect to the metric we are using.
- class 4 was almost non-existent with respect to the other classes

To cope with the imbalance of class 4 our first idea was to apply oversampling to the minority class, and one method we tried was to use patch augmentations to isolate the regions of pixels of class 4 and include them in the oversampled images. However we didn't notice any significant improvement so we later decided to handle class imbalance directly using class weights in the loss function.

To better understand the dataset, we also analyzed the noise levels and the pixel intensity distribution. Noise appeared to be moderate in variability, while we noticed relatively low-level intensity values and limited contrast.

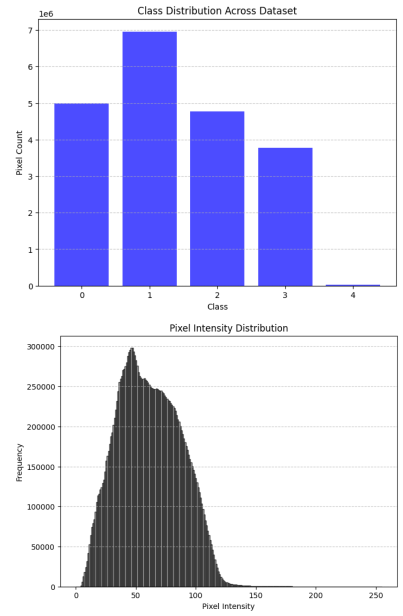

Figure 1: Class distribution and Pixel intensity distribution

3 U-net model

To perform our experiments we decided to keep 10% of samples to test our models locally and opted for a 80/20 split for training and validation.

Our initial approach was to develop a basic U-Net composed of two blocks for the Encoder, one for the Bottleneck and two for the Decoder. Each block had a stack size of two. To help the network to converge, we decided to implement the dynamic learning rate, starting with a value of 0.001. As loss function, we used Sparse Categorical Cross-Entropy, passing the background class as class to ignore during the calculation of the loss value. This choice was made because the zero class was not considered in the metric used to evaluate validation performance. As optimizer we opted for AdamW.

With this setting, we achieved a validation score of around 50%. We tried to increase both the number of blocks and the stack value both for Encoder and for Decoder, finding these results:

Configuration	mean iou (local)
4 blocks, stack=2	59,68%
4 blocks, stack=2	**61,60**%
5 blocks, stack=2	60,47%
5 blocks, stack=3	60,18%

These results were obtained by using the Add() function in the decoder blocks: on average, it brought 1% better performance than Concatenate(), so we decided to keep it for the following experiments.

4 Data pre-processing

The results of our data analysis guided us in deciding what pre-processing operations and augmentations to apply. We built some layers (see notebook for details) to combine as discussed in the Experiments section:
- Histogram-equalization layer to enhance the difference between pixel intensities by "stretching" the histogram
- Edge-extraction layer to help the net to recognize objects in the images

- Gamma-adjustment layer to experiment with darkened and brightened images, and help the Edge-extraction layer.

Since the dataset was limited in size, we decided to apply some augmentations. We wanted our augmentation pipeline to reflect realistic variations and to be guided by the dataset's own statistics, using a data-driven approach. So we tried with simple geometric transformations such as RandomFlip and RandomRotate, a RandomBrightness to act on the pixels intensity and a GaussianBlur to train the model on even lower quality images, avoiding it to concentrate on high frequency noise.

5 Experiments

We used our layers to provide our model with more versions of the same image, in order to help it in extracting features of different nature.

At first, we did this by adding or concatenating the outputs of our layers, using the result as input of the U-Net. The concatenation revealed to be better in this role.

Later, we implemented a Dual U-Net: two parallel Encoders receive two different layers' outputs and process them, then their outputs are concatenated in the input of a single Decoder.

Here we show an example, implemented with both strategies.

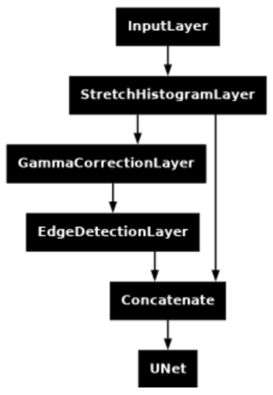

Figure 2: UNet Example

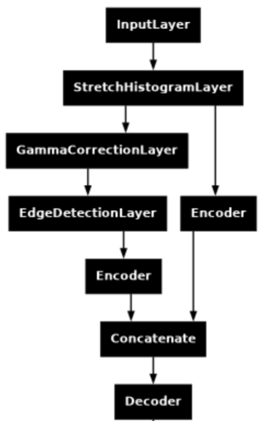

Figure 3: Dual Example

These experiments brought us some improvements, but the impact was not significant. We would have liked to try more combinations but this strategy revealed to be really time-consuming.

Anyway, with most of these combinations we managed to have mean_iou between 61 and 61,5%. Then, we had an improvement of about 0,5% for each one of them by implementing the Squeeze and Excitation block.

Eventually, we tried to implement a weighted loss function that multiply the class weights by the loss value of any generic loss function passed as a parameter. We both tried to use the class_weights obtained through the scikit learn library and a simpler manually tuned version that performed better. We also tried multiple combinations of losses such as the Dice, Focal and Boundary but somehow we found better results with the Sparse Categorical Cross-Entropy.

Even after these experiments, the best result was achieved on a U-Net model with 4 blocks and 2 stacks, compiled with a weighted Sparse Categorical Cross Entropy loss, scoring 62.324%.

6 Conclusions

We believe that our approach of starting with simple models and trying to understand how to improve them by applying different techniques gave us a good understanding of which factors can significantly benefit the model's learning. We observed that using a weighted loss function with accurate weights based on class frequencies is much more important for achieving higher performance than using overly complex and large models. One lesson we learned is that "Bigger does not mean better".

7 Further improvements

We are not satisfied with our results, but we think that our approach could bring better results with more time. Possible reasons we didn't reach excellent results are:
- We didn't focus enough on the interpretation of the predictions images
- We didn't try enough combinations of our layers.
- We didn't try enough combinations of weights and different loss in our weighted loss function.

We also would have liked to try with:
- Convolutional Attention Module
- More parallel Encoders, creating an "N-U-net", each one focused on specific characteristics

8 Contributions

Angelo Baturi	Data balancing, Dual U-Net, Weighted loss function
Daniele Bergamaschi	Data augmentation, Basic U-net, Edge detection
Lorenzo Bottelli	Data augmentation, Data balancing, Dual U-Net
Gianluca Carta	Basic U-Net, Layers, SE Block

9 References

- USE-Net: incorporating Squeeze-and-Excitation blocks into U-Net for prostate zonal segmentation of multi-institutional MRI datasets, link
- Making new layers and models via subclassing, link
- Do more with less data, link