POLITECNICO
MILANO 1863

AN2DL - Second Homework Report
Per un pugno di neuroni

Giorgio Algisi, Simone Bresciani, Ilaria Campestrini, Federico Dagani

giorgioalgisiedu, simonebresciani, ilariacampestrini, federicodagani

250859, 275432, 252865, 243989

December 14, 2024

1 Introduction

The project discussed in these pages focused on solving an *image segmentation* problem using the deep learning techniques learned during the Artificial Neural Networks and Deep Learning course.

Our objective was to develop a model that could tackle with good performances an image segmentation challenge on a Mars surface dataset.

2 Problem Analysis

We worked on an image segmentation problem that required detecting five classes (*Background, Soil, Bedrock, Sand* and *Big Rock*) in 64x128 grayscale images of Mars' rocks. To address this task, we were provided with the following datasets:

- a training set containing 2615 image-mask pairs. The **segmentation masks** were arrays with values ranging from 0 to 4, representing the five classes.
- a testing set comprising 10022 images for which we needed to predict the segmentation mask.

An analysis of the class distribution showed that the data set was quite unbalanced: the *Big Rock* class was severely under-represented compared to the others (about 1/154). This lack of representation may have distorted the model towards the majority classes, ignoring the minority.

Another important aspect we had to consider was that the *Background* was excluded from the final evaluation of the predictions. This led us to explore the option of removing the influence of this class in the loss functions and metrics.

3 Our work

3.1 Data inspection

During an initial inspection of the dataset, we had some anomalous meeting of the third-type. It seemed, indeed, that the face of an alien was stuck in some of the images. These **outliers** would have hurt significantly the performances of our model so they had to be removed from the dataset.

Moreover, upon a more thorough inspection of the segmentation masks in the training set, we realized that they were not detailed and quite rough. Indeed, some of them seemed not to follow the shape of the rocks in the images, whereas others presented an incorrect classification (often as background) of the pixels at the edges of the images. This **poor quality of the segmentation masks** could have affected the model and its performance if not addressed properly.

3.2 Data preprocessing

After a standard **normalization** of data from both images and masks to the [0,1] range, we applied more sophisticated and problem-specific data preprocessing techniques designed to address challenges that arise during the inspection of the dataset.

To **remove the outliers**, we initially attempted a hash-based approach on the images. However, due to the diversity in the size and position of the alien, this approach proved ineffective. Nevertheless, since all outliers shared the same mask, we applied a solution based on the comparison of **label hashes**, which helped identify and remove 110 samples.

To address the issue of **poor data quality** in segmentation masks, we initially attempted to make them more uniform by **replacing the spurious background pixels** in the margins with the nearest label pixels. Unfortunately, this solution did not yield the expected improvement.

As the mask modification was not successful, we decided to make our model more robust by applying the data **augmentation**. It was essential to apply it both to the images and their corresponding segmentation masks to preserve consistency between them. We implemented only horizontal flipping transformations. Other geometric or color augmentations [1] were experimented but they negatively impacted performances and we discarded it.

To address **dataset imbalance** we evaluated class weights based on class distribution. This simple approach resulted in the weight for the *Big Rock* class being set to 153.94, which was significantly higher than those of the others (0.82, 0.59, 0.84 and 1.11). We observed that this large discrepancy biased the model during training making it predicting the *Big Rock* class far too often. The reduction of its weight to 10 has helped to achieve more balanced predictions.

3.3 Model architecture

In our first attempt we implemented a **U-Net style** model by experimenting with different number of filters and layers. As shown in Table 1 (*Simple U-Net*), this first type of architecture lead to significative results, representing a good starting point.

To improve the correlations at different levels we focused on **skip connections**. Each decoder layer was densely connected to all the encoder layers using skip connections [2]. This modification lead to a huge improvement of our score (*Dense U-Net*). Different ways of combining the results of the levels have been tried: simple and weighted concatenation (with fixed and learning parameters), addition, multiplication, etc. However, these experiments did not improve performance with respect to the classical concatenation method. In addition, to enable our model to capture features at different scales, more complex structures such as **Atrous Spatial Pyramid Pooling** (ASPP) [2] and **Transformer** [5] have been introduced into the model's bottleneck.

Since the problem of combining features at different scales proved to be a crucial aspect of the model we had to design, we decided to explore a more complex architecture built by combining **two distinct U-Nets**. The first (U1) focused on fine-grained feature level information extraction, while the second U-Net (U2) was designed to capture spatial and general details. Several modes of interaction between the two networks have been tested. From a simple concatenation of the outputs of the two U-Nets working in parallel (*Parallel U-Net*), to more sequential approaches[3] with a final concatenation of the outputs of both the networks (*Dual U-Net*).

Inspired by the State-of-the-Art architectures for image segmentation, we also tried to modify our models by introducing some of their features or adopting their architecture to our problem. The main SotA architectures we took inspiration from were the *ResU-Net/++* [4], *U-Net++* [7], *Unet3+* [3], *TransU-Net* [1], *Attention U-Net* [6].

Model	Validation Mean IoU	Kaggle score
Simple U-Net	51.85±0.6	52.45±0.4
Dense U-Net	**68.60**±0.9	**70.02**±0.2
Dual U-Net	60.64±1.0	58.20±0.5
Parallel U-Net	62.65±0.3	61.13±0.7
U-Net 3+	64.23±0.2	65.79±0.1
ResNet++	59.59±0.4	58.95±0.2

Table 1: Results of the tested models

[1] Vertical flipping, contrast and brightness adjustments, ...

[2] The correspondence between different dimensions has been made possible by max-pooling and up-sampling transformations.

[3] Propagation of U1 output to U2, propagation of U1 output multiplied (pixel-wise for each element) with the initial input, skip connections between the U1 encoder and the U2 decoder, etc.

3.4 Model Loss

Much attention was given to the way of computing the loss function. Starting from the Sparse Categorical Crossentropy Loss (SSCE), we then implemented a combination of **multiple losses**. Since labeled data were scarce and noisy, our hope was that the combination of few loss functions[4] could enable the model to becomes more stable and less sensitive to label imperfections. Furthermore, we also tried to apply **dynamic weights** to the different loss functions, in order to allow the importance assigned to each loss to adapt during training. However, we were unable to find a combination that could improve the performance for each of the models we tested, leaving us with some doubts about the correctness of our approach.

Instead, we observed a significant improvement in the performance of all models when the *Background* class was excluded from the computation of the loss functions, resulting in a 10/15% increase of the score.

4 Final Model

4.1 Model Architecture

The final model architecture consisted of a **single U-Net** with **4 layers** for both the encoder and decoder (respectively with 32, 64, 128, 256 filters) and a 512-filter bottleneck. Each layer consisted of a U-Net block. We implemented the dense skip connections net of the **Dense U-Net** (introduced in Subsection 3.3). Besides, the bottleneck was a sequence of a **Transformer** block and an **ASPP** block. To prevent overfitting, the Transformer was implemented with **dropout layers** in between the Dense ones.

4.2 Training

For the evaluation of the model performances we used a combination of two metrics: **accuracy** and **Mean Intersection over Union** (Mean IoU[5]) over a validation dataset, excluding the *Background* class from the computation.

As regards loss assessment, we've decided to adopt a single static approach based on a customized **SSCE with class weights**. The main reason for choosing this solution is that it led to better performances on our final model than the others more sophisticated algorithms.

Training was conducted using *ReduceLROnPlateau*[6], a **dynamic learning scheduler** which enabled adjustments to the learning rate throughout the training epochs. Starting from 10^{-3}, it kept reducing the learning rate of a 0.1 factor every time the Mean IoU computed over the validation set wasn't improving for 40 epochs. In addition, a classical **early stopping mechanism** (with patience 50) based on the validation loss was used in order to avoiding overfitting.

5 Results & Discussion

Adopting a quite simple model enforced with some layers taken from the SotA made us reach a **70.633 final score** on Kaggle. In fact, we noticed that better results were obtained with apparently simpler architectures rather than using known SotA. That may be ascribable to the fact that most of the SotA are designed for segmentation problems on medical images that requires a precise boundary detection and small cells recognition. Instead, the Mars terrain problem we were facing, due to the given ground truth label, was more of a pattern recognition problem and so a mindful use of SotA techniques applied to a shorter path has proved to be more efficient for this specific problem.

6 Conclusions

In order to parallelize the effort of training models with different features, we worked on the project together, progressing simultaneously throughout all phases of its development.

Future work could explore using a more sophisticated dynamic learning rate, such as *CosineDecayRestarts*[7], which helps prevent the model from getting stuck in local minima.

Furthermore, a deeper analysis and experimentation on the dynamic losses function could help the training phase and get better results.

[4] Sparse Categorical Crossentropy, Dice and Boundary losses

[5] Ratio between the intersection and the union of the predicted and the true class, computed for each class

[6] tf.keras.callbacks.ReduceLROnPlateau

[7] tf.keras.optimizers.schedules.CosineDecayRestarts

References

[1] Jieneng Chen et al. "TransUNet: Rethinking the U-Net architecture design for medical image segmentation through the lens of transformers". In: *Medical Image Anal.* 97 (2024), p. 103280. URL: https://doi.org/10.1016/j.media.2024.103280.

[2] Liang-Chieh Chen et al. *Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation*. 2018. arXiv: 1802.02611 [cs.CV]. URL: https://arxiv.org/abs/1802.02611.

[3] Huimin Huang et al. *UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation*. 2020. arXiv: 2004.08790 [eess.IV]. URL: https://arxiv.org/abs/2004.08790.

[4] Debesh Jha et al. *ResUNet++: An Advanced Architecture for Medical Image Segmentation*. 2019. arXiv: 1911.07067 [eess.IV]. URL: https://arxiv.org/abs/1911.07067.

[5] Ze Liu et al. *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. 2021. arXiv: 2103.14030 [cs.CV]. URL: https://arxiv.org/abs/2103.14030.

[6] Ozan Oktay et al. *Attention U-Net: Learning Where to Look for the Pancreas*. 2018. arXiv: 1804.03999 [cs.CV]. URL: https://arxiv.org/abs/1804.03999.

[7] Zongwei Zhou et al. *UNet++: A Nested U-Net Architecture for Medical Image Segmentation*. 2018. arXiv: 1807.10165 [cs.CV]. URL: https://arxiv.org/abs/1807.10165.