POLITECNICO
MILANO 1863

AN2DL - Second Homework Report
Overfitting Gods

Bonanomi Emanuele, Dante Riccardo, Derme Francesco, Fumagalli Pietro
emanuelebonanomi, riccardodantepoli, tunamayosandwich, pietrofumagalli
278166, 274136, 274175, 279605

December 14, 2024

1 Introduction

This report illustrates the development and implementation of deep learning-based solutions to a semantic segmentation task. Several neural networks are proposed, with the goal of processing images obtained by the Curiosity Rover on Mars, distinguishing in each of them between five terrain types. The following outlines the data analysis, the attempts at overcoming criticalities and the several experiments on model architecture and training techiques.

2 Data Analysis

Of the 2615 64x128 grayscale images in the dataset, 110 were of no use: they simply consisted of uninformative noise with a nonsensical label; these outliers were internally referred to as "aliens" and **they were removed**. As highlighted by the class frequency table below (where entries are calculated as $\frac{\text{pixels in the i-th class}}{\text{total pixels}}$), there is a problem of underrepresentation with respect to the 5^{th} class:

Class	Frequency
Background	0.2431088019273952 %
Soil	0.33904544426771455 %
Bedrock	0.23277712153817365 %
Sand	0.18375470738211078 %
Big rock	0.0013139248846057885 %

This massive unbalance was the main challenge for the entire duration of the development process. Indeed, it was quite easy to recognize and segment objects of the first four classes and almost every model managed to do so, but their performance terribly dropped when it came to detecting the fifth one. Every (failed) attempt to overcome this problem is described with full details in the next section.

3 Dataset balancing

Among the techniques used to solve the problem of class imbalance the most notable were: simple augmentation, *cut-and-mix*, weighted losses, two-steps models, separated binary segmentators, and their combinations. Note that in this case simple augmentation does not refer to training-time augmentation, but rather to the process of perturbing images that specifically contained objects of the fifth class, in order to create more of them [**distorted_rocks.ipynb**]. The perturbations employed were rotations, flips and other geometric ones (resizing, translating and shearing) **accurately performed on both the samples and their masks**. In the cases were a geometric transformation created empty space in the picture, it was filled by *reflecting* the contents of the image and of its respective mask, using *bilinear* and *nearest-neighbor* as interpolation criteria respectively. In this manner, about 500 more *Big rock* samples were integrated, but the resulting dataset did not pro-

vide better results than the standard one. Another dataset was similarly enhanced through a cut-and-mix process [**cutmixed_rocks.ipynb**]: an algorithm was deployed to find *Big rock* boundaries in the original images, then these had to be painstakingly refined by hand (some pictures included multiple big rocks, in those cases the automated algorithm was not able to find boundaries by itself). These rock pieces were mixed to other samples and their labels fixed accordingly but, once again, this led to nothing. Dozens of different losses were employed to encourage the network to find big rocks. The weights were each time some sort of recalibration of those naively obtained by taking the inverse of the class frequencies; then, they were applied either to *sparse categorical cross-entropy loss*, *dice loss*, *focal loss*, *gradient magnitude loss*, *boundary loss* or various combinations, none of which could aid the model in finding big rocks.

4 Experiments

Aside from the model architecture, several techniques were played with, mainly with the goal to fight overfitting and favor generalization. For instance, a substantial train-time, batch-wise **augmentation** was performed, flipping, geometrically deforming and varying *luminosity* and *contrast* of the images. The augmentation ranges were not random. The test set was explored to find out maximum, minimum, mean and standard deviation of these variables: it was determined that they were optimally fitted by a gamma distribution with certain parameters. The samples were therefore normalized and then augmented in luminosity and contrast at test time, to match exactly the distribution of samples in the test set. This proved to be useless.

Variable	Max	Min	Mean	std	Pixel max	Pixel min
Brightness	117.8	2.6	60.749	21.723	255	1
Contrast	94.914	0.755	14.071	7.441	None	None

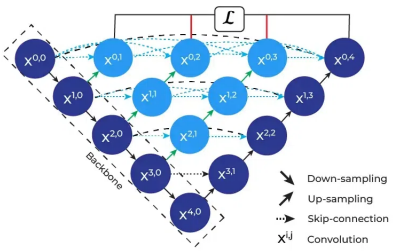
Variable	Gamma alpha	Gamma loc	Gamma beta
Brightness	12513.305	-2369.804	0.194
Contrast	3.53	-0.0898	4.012

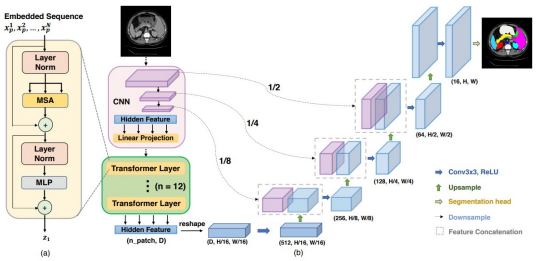
The designated optimizer was *AdamW*, and different **learning rate schedules** were tried: *warm up*, *cosine* and *exponential decay*, *reduce on plateau*. In addition to the built-in regularization of *AdamW*, *l2 regularization* was applied on convolutional layers. For a while the focus was placed on **loss functions**: trial and error led to a combination of *sparse categorical cross-entropy* (ignoring the background) and a weighted *dice loss*. During the training process, a tensor with the ground truth labels with shape (batch size, 64, 128, 1) and a tensor with the predicted ones with shape (batch size, 64, 128, 5) were passed to the loss function. In order to evaluate it, a reshaping of the first tensor was performed, passing to a representation of the true labels with one-hot-encode format. At this point the categorical cross-entropy, with the argument ignore_class set to 0 (background), was calculated and stored in a variable l. Then an element-wise product was performed between the new one-hot-encoded mask and a weight tensor (computed as $\frac{\text{total pixels}}{\text{pixel in the i-th class}}$). With this new tensor, the dice loss was calculated and stored in a second variable d. The final loss function is evaluated as $\text{loss} = d + 2 \cdot l$, and used in (almost) all the developed models.

5 Models

The recurring theme of every endeavor was the standard *U-Net*, upon which variations were introduced. The hyper-parameters to tune were the number of channels in each block and the depth of the network (in terms of reduction of spatial dimensions), in addition to the usual ones (activations, dropout rates and level of normalization). Furthermore, **skip connections** were deeply scrutinized and many possibilities were explored: from the choice between *addition* and *concatenation*, to the insertion of *residual* and *squeeze and excitation* blocks, to the complete removal of them. The list below summarize the implemented architectures:

1. The assumption that obstacles in the training process were the main hindrances to progress determined the idea of networks with **multiple intermediate outputs**. A simple structure with just two parallel upward paths (and no skip connections) ultimately produced the most successful predictions. The concept was later generalized [**multiple_outputs.ipynb**], with each segmentation head stemming from a different depth. Finally, in order to **refine the predictions**, a different Unet model was trained on the images and the corresponding generated labels. Basically, the model learned a smoothing identity: it straightened the boundaries and removed small-size regions from the input labels. [**postprocessing_model.ipynb**]





2. A *VGG16* model with uninitialized weights was used as the decoder for an otherwise standard *U-Net*: this proved to be unhelpful.

3. Two parallel networks were employed in order to let the first focus on deep features of the input images and the second on high level characteristics. The two models were trained separately and then fused at the end as they both fed a third small model whose role was just to add the two outputs, making the whole model trainable. The last layer is trained at first with the other two frozen, and then a light fine-tuning is performed with all the weights set to trainable. Regarding the two models in parallel, the first one is a classical *U-Net*, trained on a dataset where only the images that contain a big rock are augmented with random flips and rotations, and with a categorical cross-entropy as its loss. The second one is an *autoencoder*, trained with batch-wise augmentation on all the input images, and with a boundary aware loss. Fine-tuning is performed on the standard dataset and the loss described in the previous section. This model too didn't perform well on the test set.

4. Another model was produced combining a *U-Net* and a *transformer* [**trans_unet.ipynb**]. This resulted in the implementation of a network called **TransUnet**, described in [1]. The idea is to add n transformer layers within the encoder part of the U-Net: they take as input an image, already processed by a few convolutional layers, extract an certain number of squared patches from that image and assign to each an *attention score*. Those numbers are later fed into a **multi-layer perceptron** that processes and reshapes them to the size of the input image. After these layers the input images continue to flow in the classical path of the **U-Net**: bottleneck and then decoder. The training is performed with batch-wise augmentation and the usual loss. The performance of this model was way better than the others on the validation set, but it was severely overfitting and failing to generalize to the test set (despite techniques such as *weight decay*).

5. An attempt was made to train 4 different *Mobile Vit* models that would classify images in the test set that contained exclusively a single class. In order to train such models binary training labels had to be created first and to do so an image was considered to be of the class *completely background* if more than 90% of it's pixels were of that class. These models then predicted which images of the test set were completely background, completely bedrock, completely sand or completely soil; the labels of these images were then created as a solid color (for example if an image was predicted to be "completely background" then the predicted label would be fully black). To predict a label for the images that were in neither of those categories, a fifth model was trained, this was a **TransUnet** trained on a version of the dataset where all the big rocks had been replaced by background, indeed the idea was that then a post-processing model could extract the big rocks from these first unrefined labels. Following this idea a 6th model was trained on the standard dataset to take as input both the original image and the labels created by it's five predecessors, its job being that of refining these predictions and provide a final, very accurate mask. This approach failed miserably.

6 Contributions

Bonanomi Emanuele: data inspection, postprocessing, evaluation of the models;
Dante Riccardo: model (1), experiments;
Derme Francesco: augmentation, model (2) and (5);
Fumagalli Pietro: model (3) and (4), report structure.

References

[1] P. Sayak. Mobilevit: A mobile-friendly transformer-based model for image classification. 2024.