AN2DL - Second Homework Report
aaa000

Kamil Hanna, Giorgio Negro, Enrico Tirri

kamilhanna, giorgionegro, enti243

10932895, 10766662, 10742316

July 4, 2025

1 Introduction

The objective of this homework is to develop a segmentation classifier using Convolutional Neural Networks:

- The classifier must accurately classify 64x128 greyscale real images from Mars terrain into five classes, each representing a particular type of terrain.
- This is a semantic segmentation problem, meaning the classifier must assign one label to each mask pixel.

Throughout the project, the group's primary objectives have been:

- **Analyzing and Exploring** the provided dataset
- **Designing** a structured problem-solving pipeline
- **Testing and Evaluating** various approaches to address the problem
- **Enhancing** the performance of the most promising models

2 Problem Analysis

The initial dataset contains 2615 samples.

Upon examination several outliers were identified whithin the dataset as shown in figure 1

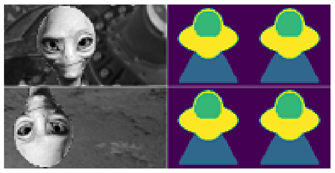

Figure 1: Sample outliers (and their labels) in training set

After removing these outliers (identified via tsne analysis) the *cleaned dataset*consist of 2505 samples. The obtained dataset has an uneven class distribution as shown in figure. 3

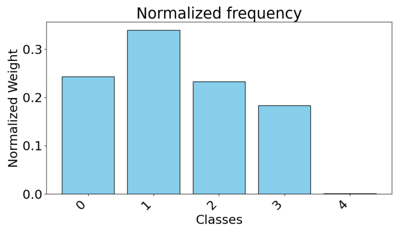

Figure 2: Normalized class distribution of the cleaned dataset.

We Initially tested without augmentation and achieved good results. Later, we introduced geometric distortions, brightness/contrast changes, embossing, and sharpening. However the performance gain didn't justify the extra training time, so we reduced augmentations to 4-6 per sample.

To address class **4** imbalance, we designed the training and validation pipeline as follows: class 4 images were separated, and the validation set included 10% of class 4 and 20% of non-class-4 images. For training, we applied 4–6 augmentations to non-class-4 images and 24–36 to class 4 images, combining originals and augmentations.

The class distribution in training set is still unbalanced after augmentation but slightly better than the initial state. 3

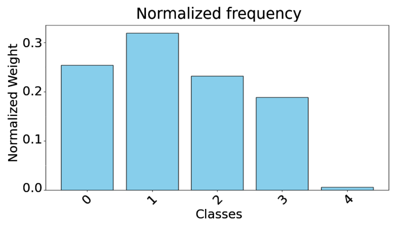

Figure 3: Normalized class distribution of the training dataset

3 Development

We started with a simple 3-depth U-Net, using batch normalization in each block and sparse categorical cross-entropy as the loss function, without any augmentation. Results were good for such a simple model 1, so we explored augmentation to improve generalization, but it didn't help. We quickly shifted focus to improving the network itself, such as:

- Add deeper layer
- Vary loss function
- Increase decoding path complexity, implementing structures as U-Net++[5], U-Net3+[2], MS-Unet[1] and MarsSeg[3]
- Add dropouts
- Add weighted gating mechanism for layer concatenation
- Add attention blocks into decoding path
- Add learnable de noising input filter for input image feature highlighting (as done in MS-Unet[1])
- Add Patch and Compose mechanism at input and output of network in order to process smaller portion of the image
- Add Residual Blocks and connections to U-Net for prevention of vanishing gradient to enhance feature learning in deeper architectures.

3.1 Experimenting with architectures

Since augmentation didn't improve scoring, we realized the core focus of the task had to be architectural choices.

The MS-Unet architecture [1] seemed like a promising approach for our problem due to the following reasons:

- We needed to process greyscale images containing many elements with fragmented shapes (e.g., rocks).
- MS-Unet demonstrated strong performance in Magnetic Resonance Elastography, which also involves greyscale images with fragmented segmentation.

Unfortunately, many aspects of the MS-Unet architecture required custom network components and forwarding mechanisms, making the complete architecture beyond our capabilities. Instead, we drew inspiration from the general structure (skip connection network and block placement) and the input image denoising method. The results were promising (see Table 1), but they were still far from the reference results reported in [1].

The same issue and results occurred with the MarsSeg architecture.

Taking inspiration from Unet+ and Unet3+ was much simpler. Although the results were still not comparable to those reported in [2], key details like the custom loss function and the classification-guided module were not implemented, and they may have been crucial for achieving better results.

3.2 Experimenting with Residual Blocks

In an attempt to improve the accuracy of our model in the early stages, we experimented with a Deep Residual U-Net architecture from [4].

Such model, introduces Residual blocks in the middle layer of the basic U-Net architecture. These Residual blocks aim to solving the vanishing gradient issue by allowing information to skip certain layers and be directly added to later layers. This leads to the ability to train very deep models without a considerable loss in learning efficiency. However the model was producing high loss values and the achieved accuracy was not the best.

3.3 Experimenting patch decomposition

Patch decomposition seemed promising, but the results were not positive. While the model correctly classified most patches, the lack of context led to poor performance in distinguishing elements of the image. We believe this behaviour stems from two key factors: the low accuracy of the training set labels and the small size of the images.

An interesting compromise could have been combining patch decomposition with full image processing [1] [3]. However, the complications introduced by such an approach led us to prefer further exploring other architectural variations.

4 Method

As the space of possible improvement was very large, we started with testing single changes to our base model. After a first round of tests we find out that the only single change that was leading to significant improvement was the substitution of batch normalization. Other modifications didn't work alone, making optimization harder as we had to test combinations of changes, which exponentially increased the search space.

4.1 Architecture

Our best submission has been obtained with a U-Net with the following characteristics:

- **Input Layer:** Takes 64x128 images with single channel (greyscale)
- **U-Net 3+ backbone:** without pre-trained weights[2]
- **U-Net block:** composed by a `Conv2D` layer, a normalization layer and an activation layer
- **Dropout layers:** placed at output of unet blocks of deep layer to prevent overfitting
- **Trainable filters:** placed at skip connection concatenation block, in order to allow trainable layer mixing in decoder stages.
- **Squeeze and excite blocks:** placed into decoding stages in between of concatenation block and unet block to enhance feature extraction.

4.2 Training

Training phase has been particularly challenging due to many main factors:

- Lack of correlation between validation result and test result
- Lack of correlation between architecture changes and validation result
- Very long training sessions
- Large number of tested architecture combination and loss functions

Model Name	Train. MeanIOU	Test MeanIOU	Train. Loss	Val. MeanIOU	Val. Loss
U-Net	0.7041	0.46656	0.7422	0.4185	0.7251
MS-Net	0.5986	0.53236	0.3467	0.6139	0.3579
Unet 3++ (best)	0.9505	0.64337	0.2028	0.7025	0.4264

Table 1: Comparison of model performance metrics.

We also struggled to project the features learned from the training set onto the validation set, as we were never able to over fit the validation set before the training set during any training session.

Moreover, due to very long training sessions and limited computational resources, most sessions were either interrupted before completion or intentionally stopped when training and validation metrics showed unsatisfactory results. Model checkpoints were periodically saved based on the best validation `mean intersection over union` value and were later used for testing.

Our best result architecture has been trained using the augmented dataset and a custom weighted sparse categorical loss function. We used custom class weights instead of dataset-based ones, which were too unbalanced and caused many false positives for class 4. The custom weights assigned zero to class 0 (excluding it from predictions), balanced weights to classes 1–3, and a 5x boost for class 4 compared to classes 1–3.

We used `AdamW` as optimizer, starting with a learning rate of 10^{-3}, which was reduced on a plateau based on the training loss.

All `Conv2D` layer implement a `l2 kernel regulizer` with a factor of 10^{-4} to stabilize predictions.

The model was trained for around 200 epochs across two sessions due to an unexpected timeout of computing resources. Each session lasted about 100 epochs, and the training state was transferred between sessions to ensure continuity.

The training dataset was processed in batches of size 64. Tests showed that doubling the batch size slightly improved model performance by about 1%. However, due to computational limitations, we couldn't use larger batch sizes with bigger models, which further constrained the potential of our approach.

5 Results

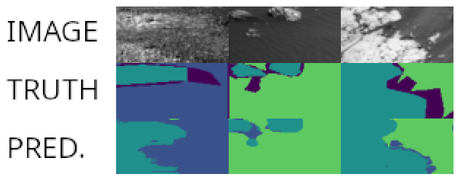

Figure 4: Samples of prediction of the validation dataset, violet color represent background class that is excluded from scoring

Our best model was able to achieve a `meaniou` score of 0.64337 on Kaggle leaderboard 1. This outcome was quite disappointing, as the model achieved a validation `meaniou` score of 0.7025 and a training `meaniou` score of 0.9505 (on the augmented dataset). Moreover, the predictions appeared highly accurate, as shown in figure 4

6 Conclusions

Results shown large space for improvement, first of all finding the a training settings that allow for a smoother learning and overfitting of both training and validation set, in order to allow meaningful test of architecture variation.

Moreover, since the accuracy of the shape of the predictions didn't seem to significantly impact the final score, as shown by the results and the initial dataset, variations in the network could be used to produce sharper and more generalized predictions, which might, counter-intuitively, lead to an increase in the score.

All `Conv2D` layer implement a `l2 kernel regulizer` with a factor of 10^{-4} to stabilize predictions. To avoid overfitting, we considered using higher values, but his would have increased the training time, witch we could not accommodate due to resources constraints.

Access to more powerful computational resources could also improve the model's performance by allowing for larger batch sizes and increased network complexity.

6.1 Contributions

We all worked in parallel on homework by employing a task-splitting strategy. In the first few days, our focus was on data preprocessing and experimenting with the basic U-Net Model. **Giorgio** was applying augmentations on the dataset, while **Enrico** was focusing on deep cleaning it. **Kamil & Giorgio** were also experimenting with the basic U-Net architecture. From there we started having regular daily mini-meetings to discuss the progress and plan the next steps.

As we move forward, **Kamil & Giorgio & Enrico** were exploring different approaches by identifying which augmentations made more sense and experimenting various modifications to the U-Net architecture. Subsequently, different architectures and many potential improvements mentioned in papers were also tested, as mentioned in the experiments part of the development section.

Finally, **Enrico** was able to produce the model with the highest accuracy. Each one of us was running the models locally and on Colab consecutively in order to make the most out of our limited hardware resources.

References

[1] H. Chen, Y. Han, P. Xu, Y. Li, K. Li, and J. Yin. Ms-unet-v2: Adaptive denoising method and training strategy for medical image segmentation with small training data, 2023.

[2] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu. Unet 3+: A full-scale connected unet for medical image segmentation, 2020.

[3] J. Li, K. Chen, G. Tian, L. Li, and Z. Shi. Marsseg: Mars surface semantic segmentation with multi-level extractor and connector, 2024.

[4] Z. Zhang, Q. Liu, and Y. Wang. Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters*, 15(5):749–753, 2018.

[5] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. Unet++: A nested u-net architecture for medical image segmentation, 2018.