AN2DL - Second Homework Report
Team Name

Denis Gabriel Sanduleanu, Alessio Caggiano, Mattia Repetti, Jonatan Sciaky

denis2404, falcro, mattiarepetti1, jonatansciakyy

251564, 258744, 251626, 212155

December 14, 2024

1 Introduction

In this project, we tackled the problem of segmenting images of Martian terrain by developing a semantic segmentation model. Our goal was to create a neural network capable of accurately segmenting unseen images into distinct classes: Background, Soil, Bedrock, Sand, and Big Rocks. To achieve this, we designed a **U-Net architecture** from scratch, focusing on effectively learning the features needed for precise segmentation. After cleaning the dataset we had around 2,500 images and their corresponding masks, enabling the training and evaluation of our model.

2 Problem Analysis

During the problem analysis phase, we examined the dataset and removed outliers to ensure data consistency. Analyzing the class distribution, we found that most classes occupied approximately 20% of the dataset area, except for the Big Rocks class, which accounted for just 0.13%. This revealed a **significant class imbalance** that needed to be addressed. The main challenges were the dataset's small size and the lack of transfer learning options, requiring our U-Net to learn solely from this limited data. To overcome this, we implemented **geometric data augmentation** while carefully preserving the integrity of masks to maintain recognizability. For the class imbalance, we assumed that careful augmentation strategies and mechanisms to prevent overfitting would be essential for training an effective model.

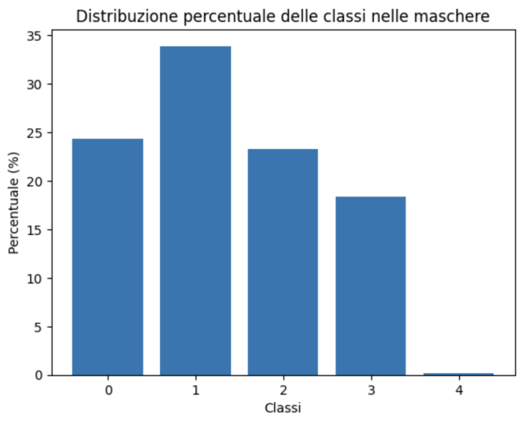


Figure 1: Class Distribution

3 Method

Our approach began with the design of a **U-Net architecture** tailored to the problem of segmenting Martian terrain. We started with a simple U-Net as the baseline and then systematically refined it to address the challenges of the dataset. Key parameters, such as **dropout rates** and **regularization terms**, were carefully adjusted to balance model capacity and generalization.

In addition to refining the baseline model, we explored several more complex approaches, employing a trial-and-error process to determine what techniques were effective. This iterative process allowed us to focus on implementing only the methods that consistently yielded better results.

Our methodology also included a detailed examination of the model's predictions after training. By analyzing these predictions, we were able to identify specific areas where the model was underperforming. This approach gave us insights into the model's weaknesses, enabling us to make targeted improvements. These iterative refinements, informed by careful observation of the model's outputs, led to major improvements in performance and more consistent results across both validation and test datasets.

4 Experiments

After augmenting the dataset using **geometric transformations**, such as **horizontal flip**, **vertical flip**, and **safe rotate**, we experimented with more complex techniques such as **CutMix** from the **Albumentations library**, we implemented a **foundational UNet architecture**. This network was designed with three **encoding levels**, a **bottleneck**, and **three decoding levels**, utilizing **skip connections** to effectively preserve spatial information. To enhance the model's generalization and address potential overfitting, we incorporated **dropout layers**, including both **standard** and **spatial dropout**.

Building on the **foundational UNet** architecture, we explored several enhancements to refine feature extraction and representation. These included modifications to the bottleneck design and the integration of advanced components such as **attention mechanisms**, **squeeze-and-excitation blocks**, **residual blocks**, **Atrous Spatial Pyramid Pooling (ASPP) modules**, and **inception blocks**. These additions aimed to improve the model's capacity to **capture multi-scale features** and complex patterns.

We also evaluated various loss functions, including **Dice**,**Weighted Sparse Categorical Crossentropy**, **Tversky**, and **Focal Tversky**, to address class imbalance and emphasize hard-to-segment regions. Recognizing the underrepresentation of certain classes in the dataset, we complemented these efforts with targeted data augmentation strategies to generate more samples for the minority class. This comprehensive approach tackled class imbalance both at the dataset level and through the design of the loss functions, ensuring better optimization for precise segmentation.

One notable iteration was the development of a **"Full-Inception UNet"**, where **inception blocks** were incorporated into the encoder, bottleneck, and skip connections to enable feature extraction at multiple scales within each layer. Despite its potential, this design did not yield the expected improvement in segmentation performance. We also experimented with extending the model's training duration by adjusting key parameters for early stopping and learning rate scheduling. Specifically, we increased the early stopping patience from 10 to a range of 20–40 epochs, allowing the model additional time to converge. Similarly, we extended the patience of the learning rate scheduler from 5 to a range of 10–15 epochs to provide a more gradual adaptation of the learning rate during training. A pivotal breakthrough came when we revisited the treatment of the background class in the loss function. **Excluding the background class from the loss computation** proved to be a key refinement, leading to a substantial jump in performance. Implementing this adjustment with the **Weighted Sparse Categorical Crossentropy loss function**, we observed a marked improvement in results, which validated the robustness of our framework and reinvigorated our efforts. This discovery underscored the importance of iterative experimentation and attention to detail in achieving our final success.

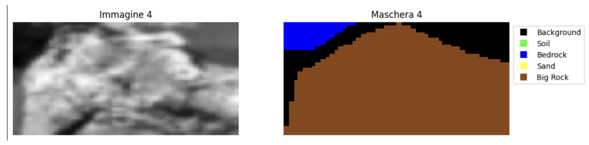


Figure 2: Specific Augmentation On Class 4

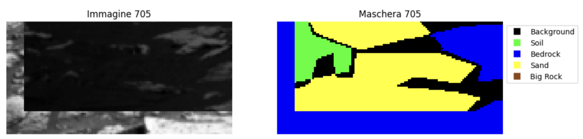


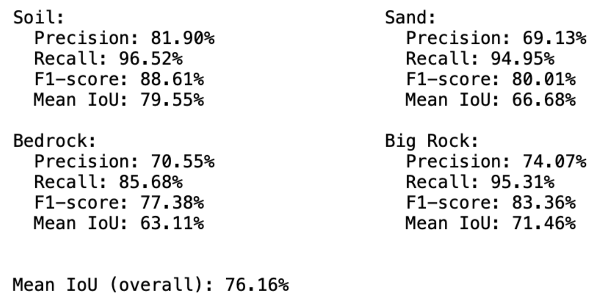
Figure 3: Augmentation With Cutmix

5 Results

We began with a simple **U-Net architecture**, achieving a mean IoU of 45% on the test set. Initially, however, the local validation set showed a much higher mean IoU of approximately 70%, indicating significant overfitting. To address this, we introduced several modifications, including the addition of **squeeze-and-excitation blocks** and **spatial dropout**, which effectively reduced overfitting and improved our test set performance to a much higher 51% IoU.

Early in our process, we identified that the big rock class was significantly underrepresented in the dataset, with only 62 images containing this class. While we applied **targeted data augmentation** for the big rock class to partially address this imbalance, much of our focus during this phase was on exploring more complex architectures and advanced techniques to improve overall performance. Despite these efforts, including trials of sophisticated models and enhancements, we eventually reached a plateau, where additional changes often degraded performance rather than yielding further gains.

Upon re-evaluating our approach, we identified a critical issue: the improper handling of class 0 during training. Initially, we excluded class 0 only from the mean IoU metric, but further analysis revealed that since the test set also ignored class 0, it would be beneficial to exclude it from the loss calculation as well. Once we implemented this change, we overcame the performance plateau that had previously hindered our progress, achieving a significant improvement to 67% mean IoU. This breakthrough reinvigorated our efforts, leading us to experiment with various new techniques, ultimately pushing the performance further to 71,13%.



```
Soil:                               Sand:
  Precision: 81.90%                   Precision: 69.13%
  Recall: 96.52%                      Recall: 94.95%
  F1-score: 88.61%                    F1-score: 80.01%
  Mean IoU: 79.55%                    Mean IoU: 66.68%

Bedrock:                            Big Rock:
  Precision: 70.55%                   Precision: 74.07%
  Recall: 85.68%                      Recall: 95.31%
  F1-score: 77.38%                    F1-score: 83.36%
  Mean IoU: 63.11%                    Mean IoU: 71.46%


Mean IoU (overall): 76.16%
```

Figure 4: Metrics Per Classes On Local Test Set

6 Discussion

The model's main strengths lie in its ability to generalize well across all classes, particularly its improved performance in predicting class 4, which was initially underrepresented. This improvement is attributed to the application of targeted augmentations and the use of a weighted sparse categorical cross-entropy loss function.

However, the model has a notable limitation: as specified earlier, it does not predict any label corresponding to the background class.

7 Conclusions

This challenge allowed us to understand how neural network development is carried out, exploring different networks and blocks, and gaining insight into the fundamental aspects of developing a neural network for segmentation. In general, the entire group participated uniformly in the various stages of development, both in terms of dataset augmentation and network development. More specifically: Jonatan worked on augmentations, creating an adequate training set, and setting up the network; Denis focused on different networks and architectures, integrating various blocks into the network; Alessio worked on augmentations and network refinements; Mattia concentrated on testing different loss functions and other network models.

References

- Github: Python for Microscopists - Models
- Github: Python for Microscopists
- Github: Keras UNet Collection
- Reddit: Stratified Split for Semantic Segmentation
- SoftwareMill: Instance Segmentation Loss Functions
- ArXiv: Paper on Semantic Segmentation
- Github: Facebook Research - Segment Anything
- Github: U-Net for Image Segmentation