AN2DL - Second Homework Report
Hendrix

Julie Ronesen Landaas, Ole Thorrud, Trygve Myrland Tafjord,

August 11, 2025

1 Introduction

This project is focused on *image segmentation* of 64x128 grayscale images from Mars terrain, with pixels classified into five categories: *Background*, *Soil*, *Bedrock*, *Sand*, and *Big Rock*. The goal is to develop a deep neural network that accurately classifies these pixels. The group's goal was to finish in the **top 20%** of the leaderboard and gain a solid understanding of image segmentation. To achieve this, we refined a baseline model using techniques from lectures, then researched and implemented additional tools to improve it.

2 Problem Analysis

Initially a data inspection was performed, revealing the following aspects of the dataset:

- The masks were of lower quality, often misrepresenting the intended segmentation.
- Outliers were present, including images of aliens, all sharing the same mask.
- Some classes were over-represented in the masks compared to others.

The key challenge was creating a model that generalizes well when trained on a suboptimal dataset. Initial assumption were that the test set does not contain lower quality masks, and that the class balance is similar to the training set class balance.

3 Method

Initially the outliers containing aliens were removed. Regarding the poor mask quality, two approaches were evaluated. The first approach involved manually discarding elements where the masks were visually misaligned or unrepresentative of their corresponding images, which as a consequence led to a reduction in the training set size. The second approach consisted of implementing a custom algorithm for improving mask quality. The algorithm iteratively expands segmented areas by examining circular neighborhoods around each pixel. If the intensity variation within a neighborhood is below a predefined threshold, the surrounding pixels are reassigned to the area's class. For all initial testing, a baseline model was created.

The baseline model employs a U-Net architecture. It features an encoder with convolutional blocks containing 32, 64, and 128 filters, each followed by max pooling for downsampling. The decoder mirrors the encoder with basic upsampling and concatenation layers to reconstruct the segmentation maps. Each convolutional block comprises three Conv2D layers with batch normalization and ReLU activations. The final layer applies a softmax activation for multi-class classification. We used sparse cross-entropy for the loss function. A custom `MeanIntersectionOverUnion` metric excludes the background class to accurately evaluate performance in accordance with test method. Training incorporates early stopping based on validation accuracy to preserve the best model weights.

The model was further improved by testing different architectures, loss functions, augmentations and blocks. During testing, long training times and limited resources constrained the number of tests the group could conduct. It was assumed that superior models would perform better across both large and smaller training sets. Consequently, a smaller training size (`test_size=0.7`) was used for much of the fine-tuning and experimentation.

4 Experiments

The experiments involved testing various model architectures, including different U-Net variations and alternative architectures. Custom loss functions

were designed, tuned, and compared, with cross-entropy as the reference loss function. Additionally, several data augmentation techniques were applied to improve model performance. Squeeze-and-excitation attention blocks, pyramid pooling, and gating mechanisms were also explored to further enhance the models. The results of these experiments are summarized in Table 1. All models listed in the table were trained using large datasets.

5 Results

The algorithm developed to enhance the masks quality worked as expected, and produced masks that were visually more descriptive of the related image. However, the baseline model trained on the altered masks did not yeld performance increases, as seen in Table 1). Additionally, removing elements where masks are deemed visually unrepresentative of their corresponding images also resulted in a decrease in performance relative to using the standard dataset. As a result, these approaches were abandoned and were not included in the final model. This suggests that the groups initial assumption that the test set contains higher quality masks was wrong, and that the model is dependent on the noisy masks to be able to generalize well.

As expected, the addition of an augmentation pipeline greatly increased performance. The initial dataset containing 2615 elements was augmented and increased up to 32064 images, leading to one of the greatest performance increases. For the baseline model, this gave a **9%** test-MIoU increase.

After researching alternative model architectures, DeepLabv3+ was identified as a strong candidate architecture to U-Net [1]. Despite this, the model did not perform well in our experiments. Research on the matter suggests that the mediocre performance could be the result of the unbalanced characteristics of the dataset [4].

Adding an extra layer and more filters to the baseline increased the performance, as seen in Table 1. Further more, replacing the standard U-Net block in the bottleneck with a Squeeze-and-Excitation block gave a performance increase of **3%**. This includes a channel attention mechanism, helping the bottleneck extract the most important features to carry through to the decoder [2]. Additionally, a gating mechanism was added to improve feature fusion in the skip connections, letting the model itself decide the optimal weight for each feature. This was attempted both with addition and concatenation, where concatenation gave the best result. Lastly we introduced transpose convolution for up-sampling instead of using the basic Upsampling2D layer to add learnable weights also in this phase. All these changes improved the baseline model, although not as much as expected.

In addition, we tried adding more advanced features like an attention mechanism [5] and pyramid pooling [3] after reading papers suggesting these could boost performance. Unexpectedly, the group did not see any increase in performance for our model.

For the loss function, we experimented with different combinations of Cross-Entropy Loss, Dice Loss, Focal Loss, and Boundary Loss. We ended up still relying heavily on the Cross-Entropy loss, but the inclusion of boundary loss improved the predicted masks' adherence to boundaries, while Dice Loss and Focal Loss helped with class imbalance and set a particular focus on the images that were hard for the model to segment, respectively. With our final choice of loss weights, we observed a difference of approximately **7%** on the test set compared to a pure Cross-Entropy loss.

The best model ended up scoring **58%** MIoU. This enhanced model is based on a U-Net structure augmented with squeeze-and-excitation blocks to emphasize important feature channels. Its encoder uses multiple convolutional blocks, each followed by max pooling, while the decoder mirrors this arrangement with upsampling and concatenation operations. With channel-wise attention applied at each stage, it focuses on critical features. The model training and validation MIoU can be seen in 1.

6 Discussion

From our results, we can see that we were not able to get a better MIoU than 0.58, even though we tried various changes in all parts of the pipeline. Compared to our baseline model with a MIoU of around 0.40, this is not as big of an improvement that we hoped for.

One major limitation in improving model performance was the restricted availability of computational resources, which constrained testing. As a result, the group could not conduct as rigorous or ex-

Table 1: Experiments and their results. Best results are highlighted in **bold**.

Model	Evaluation
Baseline without augmentation	0.40351
Baseline with improved masks	0.39320
Baseline with bad masks/images removed	0.38503
Baseline with augmentation	0.49051
Larger model (extra layer and number of filters: 64, 128, 256, 512)	0.51108
Attention U-Net	0.50255
Deeplabv3_plus	0.46136
Baseline with Squeeze and excitation (SE) bottleneck	0.51334
Larger model with SE bottleneck	0.53172
Skip connection: Gating mechanism using addition	0.48597
Skip connection: Gating mechanism using concatenation	0.51401
Larger model + SE bottleneck + pyramid pooling	0.50741
Flagship	**0.58062**

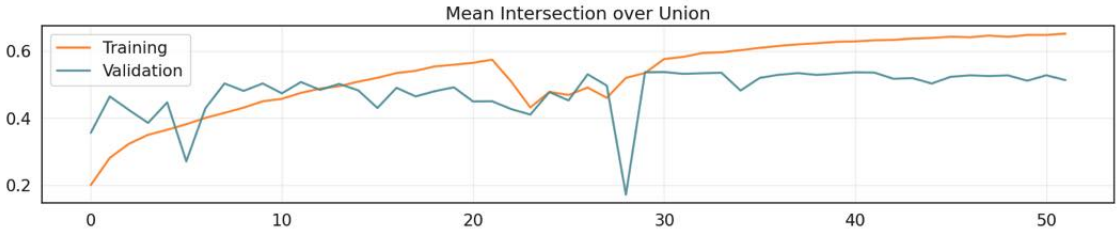

Figure 1: Training and validation MIoU for the best model

tensive experimentation as initially planned. While no substantial evidence emerged to challenge the assumption that models performing well on large datasets would also excel when fine-tuned on smaller subsets, this approach introduces potential risks. Ideally, all tuning and testing would have been conducted on large datasets to ensure the reliability and generalizability of implemented modifications, as smaller datasets may not fully capture the intricacies of the task or the model's potential

7 Conclusions

Our final model achieved a mean IoU of 0.58. Although not reaching our goal of being in the top 20%, the group gained a valuable insight into image segmentation. Further work could include improving class balance by augmentation and exploring Transformer models.

References

[1] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018.

[2] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu. Squeeze-and-excitation networks. *arXiv*, 2019.

[3] J. V. Hurtado and A. Valada. Chapter 12 - semantic scene segmentation for robotics. In A. Iosifidis and A. Tefas, editors, *Deep Learning for Robot Perception and Cognition*, pages 279–311. Academic Press, 2022.

[4] Z. Nie, J. Xu, and S. Zhang. Analysis on deeplabv3+ performance for automatic steel defects detection, 2020.

[5] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert. Attention u-net: Learning where to look for the pancreas, 2018.