AN2DL - Second Homework Report
DreamTeam

Alisa Pesotskaia, Safuan Iusupov, Davide Paltrinieri, Dario Napolitano

allythe, safuanlip, dpaltrinieri, darionapolitanoreal

272533, 272547, 276139, 275560

February 15, 2025

1 Introduction

This project targets *semantic segmentation* of Martian terrain using **deep learning** techniques.

Our approach began with **data inspection** to understand the dataset and its features. We addressed class imbalance and limited training data by experimenting with various architectures, loss functions, and augmentation strategies. Additionally, it wasn't allowed to use pretrained models.

2 Problem Analysis

2.1 Dataset

The dataset comprises **2,615** grayscale training images with corresponding label masks and **10,022** test images without masks, each with a resolution of 64×128 pixels. The provided training label masks classify Mars terrain into **five** distinct categories representing various surface types.

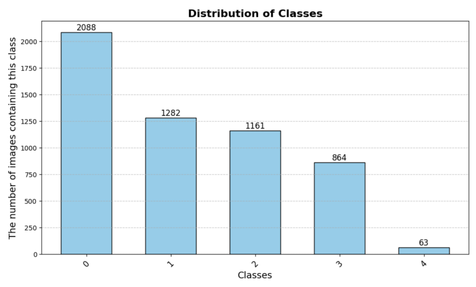

Figure 1: Class Distribution

To explore the dataset, we visualized sample images from each class and analysed the class distribution. As shown in Figure 1, a significant **class imbalance** was observed.

Additionally, we discovered that the training segmentation masks were labeled inconsistently, which we believe was due to the challenging nature of labeling Martian terrain and the involvement of multiple annotators. When annotators disagreed on the classification of a particular terrain, it was labeled as background as reported in [1].

2.2 Method

The models were evaluated using the *Mean Intersection over Union* (mIoU) metric (Equation 1).

$$\text{mIoU} = \frac{1}{|C|} \sum_{c \in C} \frac{\mathbf{1}(y = c) \wedge \mathbf{1}(\hat{y} = c)}{\mathbf{1}(y = c) \vee \mathbf{1}(\hat{y} = c)} \qquad (1)$$

Generally, the following techniques have been implemented for models training: data **normalization** by 255.0, conversion of input images **from 2d to 3d** arrays; 90-10% **dataset split** for training-validation; **excluded background** class in loss computation via masking predictions; **early stopping**; learning rate **scheduler** and **focal loss** function (Equation 2).

$$\mathcal{L} = -\sum_{n=1}^{N} (1 - t_n * y_n)^{\gamma} \log(t_n * y_n) \qquad (2)$$

3 First attempt

3.1 Outliers

Based on our experience from the previous challenge, we started from searching for the **outliers** in the given by Eugenio Lomurno dataset. An example of found outlier-alien is shown in Figure 2.

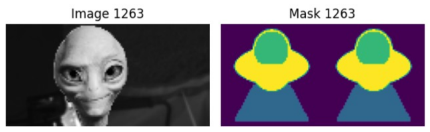

Figure 2: Example of **outliers** (image no. 1263).

After removing the outliers, the training dataset was reduced to **2,506** images, and all models were trained on this outlier-free dataset.

3.2 First Model

We trained the **U-Net++ model** with 34 million parameters implemented on keras. We also used **weighted cross-entropy loss** [2] to address **class imbalance**. The inverse of each class's pixel probability served as weights.

Despite this, the mIoU for the fourth class remained low (0.1), and the background class had disproportionately high metric (0.9). This was due to the class imbalance and a small training dataset. Our main objectives were:

- **Mitigate class imbalance**;
- **Expand the training dataset**.

4 Wrong Paths

Upon analyzing the dataset, we found that images were cropped from larger source images. We attempted to **reconstruct** them by aligning edges and key features [3], but the approach failed, as shown in Figure 3.

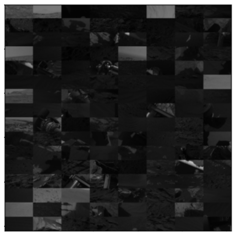

Figure 3: Example of combining crops into one image.

We also attempted to train the model without relying on segmentation masks. This approach involved addressing a classification task for small image patches (32x32) and subsequently extracting **class activation masks**. The class assigned to the patch was determined by the most frequently occurring label within it. Unfortunately, the resulting masks proved to be unreliable.

Experimenting with **losses** did not improve accuracy on validation, with losses used: boundary loss, Jaccard loss, dice loss.

We tried to implement **stacking ensemble**, which yielded predictions from three models, each generating rescaled masks (values 0–4 mapped to 0–255). These predictions were concatenated into a new sample. A new meta-model was trained on this reconstructed dataset, with the ground truth remaining the same. Unfortunately, this approach did not improve the score.

5 Experiments

During our experiments, we primarily utilized PyTorch, semantic segmentation library [4], and Albumentations library [5]. Each team member contributed to the project with their unique approach:

1. **Alisa** developed the entire training **pipeline in PyTorch**, experimented with training different **models from smp** [6] library (primarily Unet, Unet++), implemented **ensembles**, attempted to use **cam technique** and **duplicated** images containing 4 labels to solve **class imbalance**.

2. **Safuan** experimented with various **loss functions**, worked with Unet and MAnet, implemented **ensembles**, attempted to **reconstruct source images** from crops and tested different **augmentations** and image **duplication** in order to address the lack of training data.

3. **Davide** identified **outliers**, tested DeepLabV3, DeepLabV3Plus, Unet++, PSPNet and FPN, experimented with **stacking ensemble**.

4. **Dario** implemented **class balancing** through **augmenting** least represented class, analyzed and visualized the dataset and trained **LinkNet** and **Unet** models with **several encoders**.

Referring to Table 1, which displays the outcomes we reached, please note the following:

Score refers to **test mIoU** achieved during the development phase on Kaggle submissions.

Shortening	Description
Model	
DLV3, DLV3P, LN	DeepLabV3, DeepLabV3Plus, LinkNet
n+m+...	Ensemble of models at line n, m, ...
Encoder	
RN34, RN18, T, XC	ResNet34, ResNet18, timm-effnetb1, xception
MBv2, MBv3, MBs4	Mobilenetv2, Mobilenetv3, Mobileone-s4
Augmentation	
RF, FB	Random flip, RF + Brightness changes
D, DD	Duplicate images with 4th class, D + duplicate all images
RS, M	Reduce image size to 32×64, Augmentations from Albumentations library [5]
Training	
E	Ensemble
L, A	Lion optimizer, Adam or AdamW optimizer
CW, Foc(gamma)	Weighted Cross Entropy loss, Focal loss with gamma
Bound, Jac, Dice, BG	Boundary, Jaccard, Dice loss, Compute loss including background

Table 1: Summary of main experiments

#	Model	Encoder	Aug	Training	Score
1	Unet++	RN34	-	L+CW	0.476
2	Unet++	RN34	RS	L+CW	0.430
3	Unet++	RN34	FB	L+CW	0.482
4	DLV3P	RN34	FB	A+CW	0.54
5	3+4	-	-	E	0.567
6	DLV3	RN18	-	A+CW	0.450
7	DLV3	T	-	A+CW	0.390
8	Unet	MBv2	-	A+CW	0.544
9	Unet	MBv2	-	A+CW	0.546
10	Unet	MBv2	M	A+CW	0.559
11	9+10	-	-	E	0.570
12	Unet	MBv2	-	A+Foc(2)	0.548
13	Unet	MBv2	M	A+Foc(2)	0.555
14	Segformer	-	M	A+Foc(2)	0.538
15	13+10+9	-	-	E	0.585
16	MAnet	MBs4	D	A+Foc(2)	0.607
17	Unet	MBv2	D+M	A+Foc(2)	0.644
18	16+17	-	-	E	**0.660**
19	PSPnet	MBs4	D+M	A+Foc(2)	0.526
20	FPN	MBs4	D+M	A+Foc(2)	0.517
21	Unet	MBv2	D	A+Bound	-
22	Unet	MBv2	D	A+Jac	-
23	Unet	MBv2	D	A+Dice	-
24	Unet	MBv2	D	A+Foc(2)+BG	0.363
25	PSPNet	MBv2	-	A+Foc(2)	0.540
26	DLV3P	MBv3	D	A+Foc(4)	0.454
27	Unet	MBv2	DD+M	A+Foc(2)	0.502
28	Unet	X	DD+M	A+CW	0.647
29	LN	dpn107	D+M	A+CW	0.558
30	LN	dpn131	D+M	A+Foc(4)	0.586
31	Unet	XC	D+M	A+WC	0.601
32	Unet	XCv4	D+M	A+WC	0.630
33	Unet	XCrnv2	D+M	A+WC	0.647
34	31+32+33	-	-	E	0.656
35	PSPNet	MBv2	D+M	A+Foc(2)	0.636
36	Unet++	MBv2	D+M	A+Foc(2)	0.637
37	36+35+33+17	-	-	E	**0.684**

6 Results

Our main findings are as follows:

- The best model is an **ensemble** of four top-performing models, leading to a **2.4% increase in mIoU** (from 66.0% to 68.4%).
- Simple **image duplication**, focusing on the least represented class, **improved** our Unet model by 9%.
- **Including** the **background class** in the loss computation caused the model to neglect other classes, **dropping** the overall metric by more than 10%.
- **Augmentations** slightly **improved** performance but were more effective when combined with class duplication.
- **Classical architectures** like U-Net and U-Net++ with MobileNetV2 encoder **outperformed** recent models such as DeepLabV3Plus, SegFormer, PSPNet, and FPN.
- To address class imbalance, we tested various **loss functions** [2], with only *Focal Loss ($\gamma = 2$)* providing **balanced results**.
- To expand the dataset, we **duplicated all images**, but this led to overfitting, **reducing performance** by 10% compared to our best model.

7 Discussion

At times, the performance differences between models seemed inconsistent and counterintuitive. For instance, while **increasing number** of images containing 4th class **improved** the score by 9%, duplicating all the images led to overfitting and **dropped** the score.

8 Conclusions

This challenge provided valuable **hands-on experience** with semantic segmentation network training techniques, deepening our understanding of their impact. While there is still room for exploration, such as implementing the model from scratch, the process was both challenging and rewarding. This experience left us **motivated for future** competitions and eager to apply and expand these skills.

References

[1] R. Michael Swan; Deegan Atha; Henry A. Leopold; Matthew Gildner; Stephanie Oij; Cindy Chiu; Masahiro Ono; Jet Propulsion Laboratory California Institute of Technology Pasadena CA USA. "AI4MARS: A Dataset for Terrain-Aware Autonomous Driving on Mars". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2021). DOI: `10.1109/CVPRW53098.2021.00226`.

[2] Reza Azad et al. *Loss Functions in the Era of Semantic Segmentation: A Survey and Outlook*. 2023. arXiv: `2312.05391 [cs.CV]`.

[3] OpenCV. *Basics of Brute-Force Matcher*. 2024. URL: `https://docs.opencv.org/4.x/dc/dc3/tutorial_py_matcher.html` (visited on 12/11/2024).

[4] Pavel Iakubovskii. *Segmentation Models Pytorch*. `https://github.com/qubvel/segmentation_models.pytorch`. 2019.

[5] Alexander Buslaev et al. "Albumentations: Fast and Flexible Image Augmentations". In: *Information* 11.2 (2020). ISSN: 2078-2489. DOI: `10.3390/info11020125`. URL: `https://www.mdpi.com/2078-2489/11/2/125`.

[6] Pavel Iakubovskii. *Segmenation models pytorch*. 2024. URL: `https://smp.readthedocs.io/en/latest/models.html` (visited on 12/12/2024).