POLITECNICO
MILANO 1863

AN2DL - First/Second Homework Report
Johnny Deep (Learning)

Andrea Giangrande, Marta Giliberto, Emanuele Greco,
batman010, martagiliberto, lelegreco2002,
252647, 258876, 252080,

February 27, 2025

1 Introduction

The aim is to develop a **deep learning** model for **semantic segmentation** of 64x128 grayscale Mars terrain images, prioritizing accuracy and efficiency. The model will need to load a dataset and be able to correctly identify the prevalent terrain type of the image analyzed.

2 Problem Analysis

The dataset consisted of about 2,500 images and their corresponding masks, where each pixel in the images was assigned to a specific class. Images with **alien figures** were removed after data visualization. The dataset was then split into 80% for training and 20% for validation, as the test set provided was already sufficiently large. Light augmentation was applied to improve model generalization. To ensure consistency, all geometric transformations were applied simultaneously to both images and their corresponding masks. It is worth noting that the dataset was **heavily unbalanced**, presenting millions of pixels for all classes, ranging between 6 to 11 millions, except for class 4, labeled as "*Big Rock*", where barely 40,000 pixels where included. Dataset imbalance and inconsistent masks added challenges.

3 Method

The team approached the semantic segmentation task iteratively, refining the model and training process based on observed results and challenges. Initially, a simple UNet[1] model was implemented using Sparse Categorical Crossentropy[9] as the loss function. During this phase, the dataset still contained outliers, such as the alien class, which was later identified as a significant source of noise. Despite this, the initial model achieved a Mean Intersection over Union (Mean IoU) score of 0.48 on the test set. After analyzing the dataset and removing the outliers, the same model was retrained on the cleaned data, yet the performance unexpectedly dropped slightly, fluctuating around 0.44 Mean IoU.

3.1 Model Architecture Exploration

Over several iterations, various architectures and enhancements were tested to improve segmentation performance. These included:

1. **UNet Variants:**
    - A dual UNet structure combining a global and local network for capturing both contextual and fine-grained details.
    - UNet++ and UNet3+ architectures to experiment with skip connections and feature fusion.

2. **Advanced Features in the Bottleneck:**
    - Incorporation of SE (Squeeze-and-Excitation) blocks for adaptive feature recalibration.
    - Pyramid Pooling and Atrous Spatial Pyramid Pooling (ASPP) to integrate multi-scale features.
    - Parallel Dilated Convolution for enhanced multi-scale context.
3. **Custom Layers and Components:**
    - Additional downsampling and upsampling layers with attention gates to focus on relevant regions.
    - Adaptive dropout layers to dynamically adjust regularization during training.
4. **Alternative Architectures:**
    - DeepLabV3+[2] with ResNet50[5] was tested but underperformed.

Despite these changes, performance metrics hovered between 0.45 and 0.48 Mean IoU, indicating the need for further optimization.

3.2 Loss Function Experiments

Given the challenges in achieving significant improvements, the team turned to loss function experimentation:

- **Baseline Losses:** Sparse Categorical Crossentropy[9], Dice Loss, Boundary Loss and Focal Loss were evaluated individually and in combination.
- **Class Imbalance Mitigation:** Weighted loss functions and `class_weight` parameters were applied to address the significant imbalance in class 4, which had very few samples
- **Jaccard Loss:** A custom Jaccard Loss was also tested with positive results.
- **Combined Loss:** A combination of Dice Loss and Jaccard Loss was found to be the most effective

To address class imbalancing, the label 'background' was strongly penalised and also removed in some loss experimentattion, resulting in an increment of the meanIoU valutation.

3.3 Optimizer Exploration

In addition to architectural and loss function experiments, various optimizers were tested to improve convergence and performance:

- **SGD (Stochastic Gradient Descent):** Tried with different learning rates and momentum settings.[8]
- **Adam and AdamW:** Used for their adaptive learning capabilities and weight decay.[3]
- **RMSProp:** Experimented for handling non-stationary objectives.[7]

The best results were achieved with AdamW in combination with learning rate scheduling.

3.4 Data Augmentation

To enhance generalization, extensive data augmentation techniques were employed:

- Standard augmentations included random flipping, rotations, and brightness adjustments.
- The augmentation pipeline was applied in preprocessing as viewed on the exercise lessons, to spped-up the training and simplify the label mapping.

3.5 Final Model

The final architecture was based on a UNet with ResNet-like blocks for feature extraction and downsampling. Key components included:

- **ResNet-Inspired Blocks:** A residual convolutional module inspired by the ResNet design, incorporating convolutions, batch normalization, and a skip connection. These components enhance gradient flow during training, promoting faster and more stable convergence.
- **Attention Gates:** Integrated in the upsampling path to enhance focus on relevant regions.
- **Bottleneck Module:** A multi-scale feature extractor with SE blocks and dilated convolutions.

- **Adaptive Components:** Instance normalization and adaptive dropout layers for dynamic training adjustments.
- **Optimizer:** AdamW
- **Loss:** Combined Dice+Jaccard
- **Early stopping and learning rate scheduler:** To prevent overfitting and ensure convergence [4] [6]

4 Experiments

Table 1: Models tested with relative results obtained on validation and test set. All models were trained on a slightly augmented dataset, combined with the original one.

Model	Validation Mean-IoU	Test Score
Single U-Net+**Outliers**	47.45%	48.19%
Single U-Net+**Sparse**	46.45%	44.70%
Single U-Net+**Combined**	46.46%	45.95%
Single U-Net+**Jaccard**	58.85%	58.96%
ResUnet+**Focal**+**Dice**	44.00%	47.00%
ResUnet+**Dice**+**Jaccard**	64.00%	**62.30%**
ResUnet+**Jaccard**	64.43%	61.00%
Dual U-Net+**Sparse**	45.03%	41.40%
Dual U-Net+**Weighted**	43.16%	40.78%
Dual U-Net+**Combined**	43.94%	Untested

5 Results

Table 1 shows that simpler models outperformed more complex ones, possibly due to limited data and image resolution. This result keeps presenting itself when different architectures come in place, like a Double U-Net or DeepLabV3+. A positive difference was made by the **balancing** of the classes in the **loss** adopted: the combinations with this factor have been able to outperform the others.

6 Discussion

The results achieved during this project highlight both strengths and limitations of the adopted methodology. While the team successfully improved the Mean IoU from an initial 0.48 to a final 0.62 by iteratively refining the architecture, class balancing, loss functions, and data augmentation techniques, some weaknesses and assumptions limited the overall performance.

6.1 Strengths

Comprehensive Architecture Exploration: The iterative testing of various UNet variants, including advanced features such as SE blocks, ASPP, and attention gates, demonstrated the team's commitment to architectural innovation.
Augmentation Strategy: Augmentation played a smaller but still important role in improving model robustness and generalization.
Training: Training the model with multiple training cycles slightly improved the model performance. Choosing the best training parameters also played an important role.

6.2 Weaknesses and Limitations

Class Imbalance: A significant limitation was the imbalance in class 4, which had very few samples compared to other classes.
Outlier Handling: While removing the alien class improved dataset quality, initial models were trained with these outliers, which may have skewed early results.
Optimizer Sensitivity: Despite testing several optimizers, the model's performance remained sensitive to hyperparameter tuning, requiring extensive experimentation to achieve stability.

7 Conclusions

The work was evenly split across the group member: all group members worked in parallel to try and implement the best possible model, to then combine all the results and get the best features from each one.
Future work could address the limitations described in Section 6.2 by focusing on more advanced methods for handling class imbalance, such as oversampling or generating synthetic examples for rare classes, as well as exploring additional architectural innovations to further enhance performance.

References

[1] F. Chollet. Image segmentation with a U-Net-like architecture, 2020.

[2] S. Rakshit. Multiclass semantic segmentation using DeepLabV3+, 2024.

[3] K. Team. Adam Optimizer Documentation, 2024.

[4] K. Team. EarlyStopping Callback Documentation, 2024.

[5] K. Team. Keras ResNet50 Documentation, 2024.

[6] K. Team. ReduceLROnPlateau Callback Documentation, 2024.

[7] K. Team. RMSprop Optimizer Documentation, 2024.

[8] K. Team. SGD Optimizer Documentation, 2024.

[9] TensorFlow. Tensorflow Documentation, 2024.