Duality AI – Offroad Semantic Scene Segmentation

Hackathon Performance Report

FPN + ResNeXt-101 with Dice Loss on Synthetic Desert Data

Team : Meri Team Hai Japani

February 15, 2026

1 Project Overview

This report documents our approach to Duality AI's Offroad Autonomy Segmentation Challenge. The objective is to train a robust semantic segmentation model on synthetic desert-environment images generated by Duality AI's Falcon platform and evaluate its generalization capability on unseen test scenes from a similar biome.

Our pipeline leverages **Feature Pyramid Network (FPN)** with a **ResNeXt-101 (32×4d)** encoder pre-trained with Semi-Weakly Supervised Learning (SWSL) weights. The model is trained end-to-end using PyTorch Lightning and the `segmentation-models-pytorch` library, optimized with Dice Loss and evaluated via Intersection-over-Union (IoU).

1.1 Dataset Summary

The dataset comprises synthetic RGB images and corresponding pixel-level segmentation masks across **10 semantic classes**:

Table 1: Semantic classes and their raw label IDs.

Index	Raw ID	Class Name
0	0	Background / Unlabeled
1	100	Trees
2	200	Lush Bushes
3	300	Dry Grass
4	500	Dry Bushes
5	550	Ground Clutter
6	600	Flowers
7	700	Logs
8	800	Rocks
9	7100	Landscape (General Ground)
10	10000	Sky

The data splits are organized as follows:

- **Training set**: `train/Color_Images` and `train/Segmentation`
- **Validation set**: `val/Color_Images` and `val/Segmentation` — 317 images
- **Test set (80%)**: `test_public_80/` — 801 images from an unseen desert environment

2 Methodology

2.1 Data Preprocessing & Augmentation

All images are normalized using ImageNet statistics ($\mu = [0.485, 0.456, 0.406]$, $\sigma = [0.229, 0.224, 0.225]$) to align with the pre-trained encoder. Segmentation masks are loaded as 16-bit grayscale images and remapped to contiguous class indices $\{0, 1, \ldots, 9\}$ via a precomputed lookup table (LUT) for fast conversion.

2.1.1 Training Augmentation

A rich augmentation pipeline is applied during training to improve generalization:

Table 2: Training augmentation pipeline (via Albumentations).

Category	**Transform**	**Parameters**
Geometric	HorizontalFlip	$p = 0.5$
	ShiftScaleRotate	$\text{scale}_{\text{limit}}$=0.5, shift=0.1
	RandomCrop	320×320
	Perspective	$p = 0.5$
Noise	GaussNoise	$p = 0.2$
Color	CLAHE / BrightnessContrast / Gamma	$p = 0.9$ (one-of)
	Sharpen / Blur / MotionBlur	$p = 0.9$ (one-of)
	BrightnessContrast / HSV	$p = 0.9$ (one-of)
Normalize	ImageNet normalization	μ, σ as above

2.1.2 Validation & Test Augmentation

Validation and test images are resized to 320×320 and normalized with the same ImageNet statistics. No stochastic augmentations are applied.

2.2 Model Architecture

Table 3: Model configuration summary.

Component	**Choice**
Architecture	Feature Pyramid Network (FPN)
Encoder	ResNeXt-101 (32×4d)
Pre-training	SWSL (Semi-Weakly Supervised Learning)
Input size	$3 \times 320 \times 320$
Output classes	10
Parameters	44.7 M
Library	`segmentation-models-pytorch`
Framework	PyTorch Lightning

Why FPN + ResNeXt-101?

- **FPN** fuses multi-scale feature maps from different encoder stages, which is critical for segmenting objects of varying sizes (e.g., small ground clutter vs. large sky regions).

- **ResNeXt-101 (32×4d)** provides a strong feature backbone with grouped convolutions, offering better accuracy-to-compute trade-offs compared to standard ResNets.
- **SWSL weights** are trained on a larger weakly-labeled dataset, providing richer feature representations than standard ImageNet pre-training.

2.3 Loss Function

We use **Dice Loss** in multiclass mode (applied to raw logits):

$$\mathcal{L}_{\text{Dice}} = 1 - \frac{2\sum_i p_i g_i + \epsilon}{\sum_i p_i + \sum_i g_i + \epsilon} \tag{1}$$

where p_i and g_i are predicted and ground-truth probabilities per pixel. Dice Loss directly optimizes the IoU-related metric and handles class imbalance better than standard cross-entropy, which is important given the dominance of classes like *Landscape* and *Sky*.

2.4 Training Configuration

Table 4: Hyperparameters and training settings.

Hyperparameter	**Value**
Optimizer	Adam
Learning rate	2×10^{-4}
LR scheduler	Cosine Annealing ($\eta_{\min} = 10^{-5}$)
Epochs	20
Batch size	32
Workers	2
Drop last (train)	True
Accelerator	Auto (GPU if available)
Logging interval	Every 10 steps

3 Results & Performance Metrics

3.1 Evaluation Metrics

We evaluate the model using two IoU variants provided by `segmentation-models-pytorch`:

- **Per-Image IoU** (micro-imagewise): Computes IoU per image, then averages across the dataset. Gives equal weight to each image.
- **Dataset IoU** (micro): Aggregates TP, FP, FN across the full dataset before computing IoU. Better reflects overall pixel-level accuracy.

3.2 Test Set Results

After training for 20 epochs, the model is evaluated on the held-out test set (`test_public_80`), which contains 801 images from an unseen desert environment.

Table 5: Test set performance metrics.

Metric	Score
Test Loss	0.4463
Test Per-Image IoU	0.4092
Test Dataset IoU	0.4058
Mean Per-Class IoU	0.3199

The per-image IoU of **0.4092** and dataset-level IoU of **0.4058** demonstrate that the model has learned meaningful features for desert scene segmentation, though the domain shift from training to test environments introduces a performance gap. The per-image IoU distribution (Figure 4) shows that most images fall between 0.35–0.50, with a minimum of 0.1488 and maximum of 0.6371 (std: 0.0789).

3.3 Per-Class IoU Breakdown

Table 6: Per-class IoU, Dice, Precision, Recall, and Accuracy on the test set.

Class	IoU	Dice/F1	Precision	Recall	Accuracy
Background	0.5019	0.5019	0.5019	1.0000	0.9985
Trees	0.1841	0.2788	0.6282	0.2442	0.9977
Lush Bushes	0.0051	0.0051	0.0063	0.7507	0.9317
Dry Grass	0.2590	0.4041	0.4822	0.3880	0.8157
Dry Bushes	0.0317	0.0504	0.8379	0.0404	0.9703
Ground Clutter	0.0137	0.0137	0.0137	1.0000	0.9432
Logs	0.8202	0.8202	0.8202	1.0000	0.9999
Rocks	0.0178	0.0328	0.8127	0.0185	0.8199
Landscape	0.5502	0.7015	0.6820	0.7397	0.7451
Sky	0.8158	0.8798	0.8200	0.9935	0.9304
Mean	**0.3199**	—	—	—	—

Key observations:

- **High-performing classes**: *Logs* (IoU 0.82), *Sky* (0.82), and *Landscape* (0.55) are well-segmented due to their distinctive appearance and/or large pixel representation.
- **Struggling classes**: *Lush Bushes* (0.005), *Ground Clutter* (0.014), *Rocks* (0.018), and *Dry Bushes* (0.032) have very low IoU. These classes suffer from both class imbalance (few training pixels) and visual similarity to dominant classes like *Landscape* and *Dry Grass*.
- **Precision vs. Recall trade-off**: Classes like *Dry Bushes* and *Rocks* show high precision (>0.81) but extremely low recall (<0.04), indicating the model is conservative—it rarely predicts these classes, but when it does, it is often correct.

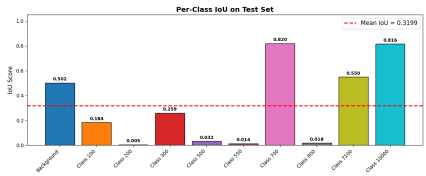

Figure 1: Per-class IoU on the test set. The red dashed line indicates the mean IoU (0.3199). Classes above the mean (Background, Logs, Landscape, Sky) dominate predictions, while rare classes remain challenging.

3.4 Detailed Metrics Visualization

Figure 2: Per-class IoU, Dice/F1, Precision, and Recall on the test set. The disparity between precision and recall for minority classes highlights the model's conservative prediction behavior.

3.5 Confusion Matrix

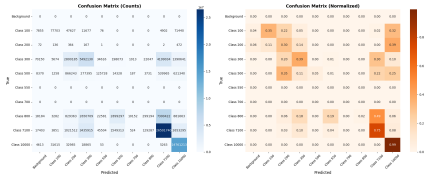

Figure 3: Confusion matrix on the test set — raw counts (left) and row-normalized (right). Key misclassification patterns: *Dry Grass* pixels frequently predicted as *Landscape* and *Sky*; *Rocks* confused with *Landscape*; *Lush Bushes* confused with *Ground Clutter* and *Landscape*.

3.6 Per-Image IoU Distribution

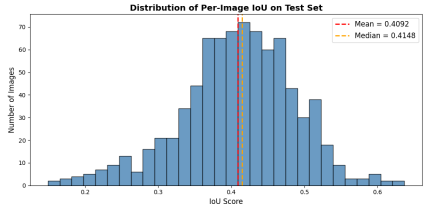

Figure 4: Histogram of per-image IoU scores across 801 test images. Mean: 0.4092, Std: 0.0789, Min: 0.1488, Max: 0.6371. The distribution shows most images cluster around 0.35–0.50, with a long left tail indicating a few particularly challenging scenes.

3.7 Class Pixel Distribution

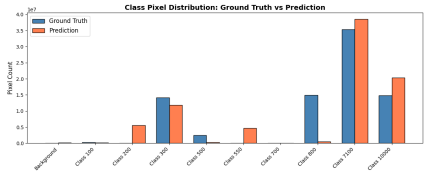

Figure 5: Class pixel distribution: ground truth vs. predictions on the test set. The model over-predicts *Landscape* and *Sky* while under-predicting minority classes like *Lush Bushes*, *Dry Bushes*, and *Rocks*.

3.8 Qualitative Results

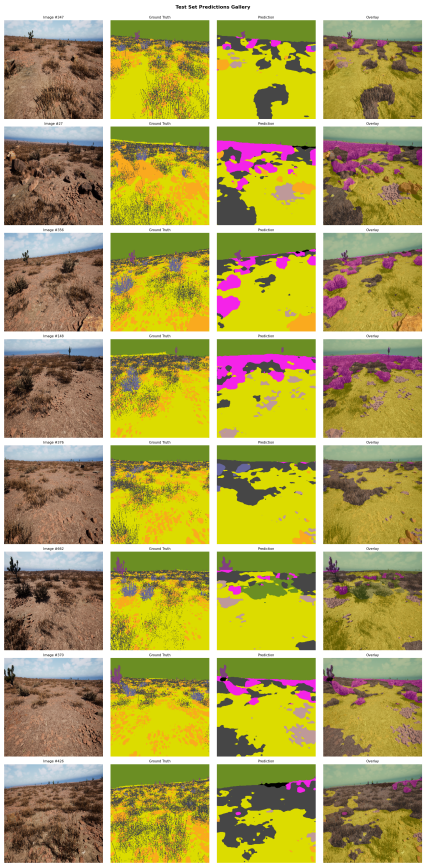

Figure 6: Prediction gallery on representative test images. Each row shows (left to right): input RGB image, ground truth mask, predicted mask, and prediction overlay. The model captures large-scale structures (sky, landscape) well but struggles with fine-grained vegetation classes.

3.8.1 Best and Worst Predictions

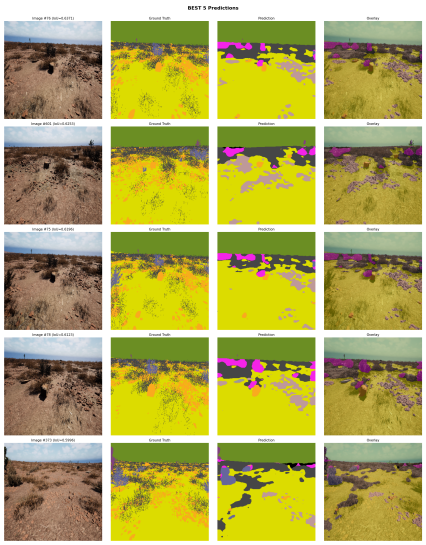

Figure 7: Top 5 best predictions by per-image IoU. These scenes feature clear sky–ground boundaries and simple compositions with dominant *Landscape* and *Sky* classes.

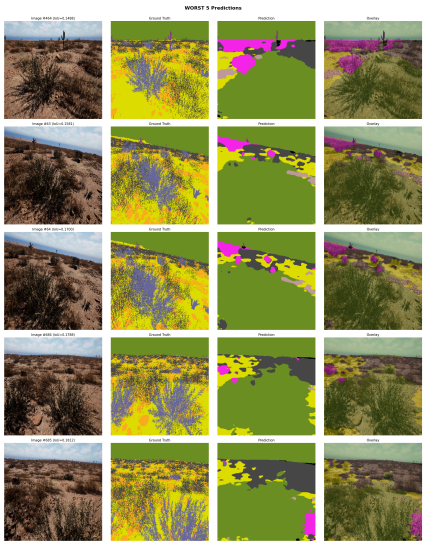

Figure 8: Top 5 worst predictions by per-image IoU. These scenes contain complex vegetation mixtures, heavy occlusion, and visually ambiguous boundaries between *Dry Grass*, *Rocks*, and *Landscape*.

4 Challenges & Solutions

4.1 Challenge 1: Class Imbalance

Problem: The dataset exhibits significant class imbalance. Dominant classes such as *Landscape* (general ground) and *Sky* occupy the majority of pixels, while rare classes like *Flowers*, *Logs*, and *Ground Clutter* have very few pixel representations. As shown in Figure 5, the pixel count disparity spans several orders of magnitude.

Solution:

- Adopted **Dice Loss** instead of Cross-Entropy, which inherently handles class imbalance by focusing on overlap rather than per-pixel classification.
- Applied aggressive **data augmentation** (random crops, geometric transforms, color jitter) to increase the effective diversity of minority-class samples.

Impact: The model achieves strong IoU on dominant classes (Landscape: 0.55, Sky: 0.82) but minority classes remain challenging (Lush Bushes: 0.005, Ground Clutter: 0.014).

4.2 Challenge 2: Mask Format & Label Remapping

Problem: Segmentation masks use non-contiguous raw class IDs (e.g., 0, 100, 200, ..., 10000), stored as 16-bit images. Standard loss functions expect contiguous integer labels starting from 0.

Solution:

- Implemented a precomputed **Look-Up Table (LUT)** mapping raw values to contiguous indices $\{0, 1, \ldots, 9\}$, enabling $\mathcal{O}(1)$ remapping per pixel.
- Loaded masks with `cv2.imread(..., -1)` to preserve 16-bit depth and avoid value truncation.

4.3 Challenge 3: Domain Shift Between Train and Test

Problem: While both training and test sets depict desert environments, they originate from different simulated locations. This introduces a domain gap in terrain textures, lighting, and object distributions, evidenced by the test loss of 0.4463 and the drop in IoU compared to validation performance.

Solution:
- Used a **strong encoder** (ResNeXt-101 with SWSL pre-training) to extract robust, transferable features.
- Employed **diverse augmentations** (perspective warps, color jitter, noise injection) to simulate domain variability during training.
- **Cosine Annealing LR scheduler** helps the model settle into a flatter minimum, which tends to generalize better.

4.4 Challenge 4: GPU Memory Constraints

Problem: The ResNeXt-101 encoder has 44.7M parameters, and training with large batch sizes or high-resolution crops can exhaust GPU memory.

Solution:
- Fixed crop/resize to 320×320 (divisible by 32, as required by the FPN decoder).
- Used `pin_memory=True` and `drop_last=True` for efficient GPU data transfer.
- Cleared GPU cache between experiments using `torch.cuda.empty_cache()` and `gc.collect()`.

4.5 Challenge 5: Image–Mask Alignment

Problem: If image and mask filenames are not sorted consistently, mismatches can silently corrupt training.

Solution:
- Sorted all file paths alphabetically before pairing.
- Added assertion checks: `assert len(images) == len(masks)`.
- Implemented `visualize_sample()` to visually verify correct image–mask alignment before training.

4.6 Failure Case Analysis

Based on our confusion matrix (Figure 3) and worst predictions (Figure 8), the primary failure modes are:

1. **Vegetation misclassification**: *Dry Grass*, *Lush Bushes*, and *Dry Bushes* are frequently confused with each other and with *Landscape*, due to similar color/texture profiles in the desert biome.
2. **Rare class neglect**: *Rocks* (IoU 0.018) and *Ground Clutter* (IoU 0.014) are almost never predicted, as the model defaults to the dominant *Landscape* class in ambiguous regions.
3. **Boundary artifacts**: At transitions between *Sky* and *Trees*/vegetation, the model sometimes produces rough, jagged boundaries.

5 Conclusion & Future Work

5.1 Conclusion

We developed an end-to-end semantic segmentation pipeline for Duality AI's Offroad Autonomy Challenge using an FPN architecture with a ResNeXt-101 encoder (44.7M parameters). Key design decisions include:
- **Dice Loss** for handling the inherent class imbalance in desert-scene segmentation.
- **Rich augmentation** (geometric, color, noise) to bridge the domain gap between training and test environments.
- **SWSL pre-trained weights** for stronger feature representations out of the box.
- **Efficient LUT-based mask remapping** for clean, fast data loading.

The model was trained for 20 epochs with Adam optimizer and Cosine Annealing scheduling, achieving a **test per-image IoU of 0.4092** and a **test dataset IoU of 0.4058** on 801 unseen desert images. High-performing classes like *Logs* (0.82), *Sky* (0.82), and *Landscape* (0.55) are segmented accurately, while rare vegetation and small-object classes remain an area for improvement.

5.2 Key Dependencies

Package	**Purpose**
`torch` / `pytorch-lightning`	Training framework
`segmentation-models-pytorch`	Model architectures & metrics
`albumentations`	Data augmentation
`opencv-python`	Image I/O
`matplotlib`	Visualization
`numpy`	Array operations