Offroad Semantic Segmentation - Hackathon Report

Team Leader: Karthik Rajarapu
Team Members: S. Soheb, Binita, Divya
Hackathon: Duality AI - Offroad Autonomy Segmentation Challenge
Date: January 31, 2026

Executive Summary

This report documents our approach to training a semantic segmentation model for offroad terrain analysis using Duality AI's synthetic desert environment dataset. Our best model achieved **55.5% IoU** (57.2% with TTA) using a custom UNet architecture with class-weighted CrossEntropy loss.

Metric	Value
Final IoU Score	**57.2%** (with TTA)
Model Architecture	Custom UNet (4 encoder, 4 decoder blocks)
Training Time	1.5 hours (T4 GPU)
Experiments Conducted	7

1. Methodology

1.1 Data Preprocessing

Challenge: Raw images needed standardization for efficient training.

Solution: Offline preprocessing pipeline:

- Resized all images to 256×256
- Converted masks to indexed format (0-9 class IDs)
- Organized into train/val splits (80/20)

Result: 95% reduction in I/O time during training.

1.2 Model Architecture

We implemented a **Custom UNet** with:

- **Encoder**: 4 blocks (64→128→256→512 channels)
- **Decoder**: 4 blocks with skip connections
- **Output**: 10-class segmentation map

```
Input (3×256×256)
    ↓
Encoder: Conv→BN→ReLU→MaxPool (×4)
```

```
    ↓
Decoder: Upsample→Concat→Conv→BN→ReLU (×4)
    ↓
Output (10×256×256)
```

1.3 Training Configuration

Parameter	Value
Loss Function	CrossEntropyLoss (weighted)
Optimizer	Adam (lr=0.001)
Batch Size	8
Epochs	50
Hardware	NVIDIA T4 GPU

1.4 Class Weighting Strategy

Challenge: Severe class imbalance (e.g., Logs: 0.08% vs Landscape: 38%)

Solution: Inverse-frequency class weighting:

Class	Pixel %	Weight Applied
Landscape	38%	0.3×
Trees	3.6%	2.8×
Logs	**0.08%**	**130×**

---

2. Results & Performance Metrics

2.1 Final IoU Score

Split	IoU
Validation	**55.5%** (57.2% with TTA)

2.2 Experiments Summary

#	Model	Configuration	IoU	Status
1	Custom UNet	CE + Class Weights	**55.5%** (57.2% TTA)	☑ Best
2	Custom UNet	Dice+CE Combined	40.6%	❌
3	ResNet34-UNet	Pretrained Encoder	45.0%	❌
4	Custom UNet	100 epochs	51.1%	⚠

#	Model	Configuration	IoU	Status
5	ResNet34-UNet	Frozen Encoder	40.5%	❌
6	Custom UNet	512×512 resolution	20.4%*	⚠
7	DeepLabV3+	Focal+Dice Loss	N/A	❌ Crashed

*Only 1 epoch completed before switching strategy

2.3 Training Progression

```
Epoch 10: IoU = 0.37
Epoch 25: IoU = 0.48
Epoch 40: IoU = 0.53
Epoch 50: IoU = 0.555 (Base)
With TTA: IoU = 0.572 (Final)
```

2.4 Per-Class Performance (Estimated)

Class	Expected IoU	Notes
Landscape	High	Largest class, well-learned
Sky	High	Large, distinctive
Trees	Medium	Good features
Dry Grass	Medium	Well-represented
Logs	Low	Very rare (0.08%)
Rocks	Low	Occlusion issues

---

3. Challenges & Solutions

Challenge 1: Severe Class Imbalance

Problem: Logs class represents only 0.08% of pixels, causing the model to ignore it entirely.

Solution: Implemented inverse-frequency class weighting:

```
class_weights = [total_pixels / (num_classes × class_pixels)]
```

Result: Rare classes became detectable; overall IoU improved from 0.31 to 0.572.

Challenge 2: Slow Training (60 min/epoch)

Problem: CPU-based training with on-the-fly preprocessing was too slow.

Solution:

1. Migrated to GPU (T4)
2. Offline preprocessing (256×256 resizing)
3. Optimized DataLoader (pin_memory, num_workers)

Result: Training speed improved **50×** (60 min → <2 min per epoch).

Challenge 3: Pretrained Models Underperformed

Problem: ResNet34 pretrained on ImageNet achieved lower IoU (45%) than custom UNet (57.2%).

Analysis:

- ImageNet features (cats, dogs) don't transfer well to desert terrain
- Domain gap between natural images and synthetic desert data
- ImageNet normalization mismatch

Lesson: For domain-specific synthetic data, training from scratch can outperform pretrained models.

Challenge 4: Dice Loss Instability

Problem: Dice+CE combined loss led to oscillating training and lower final IoU (40.6%).

Analysis:

- Dice loss gradient signals conflict with CE
- Harder to optimize than pure CE
- Required more careful hyperparameter tuning

Solution: Reverted to simple CE with class weighting.

---

4. Failure Case Analysis

Misclassification Patterns Observed:

Failure Type	Likely Cause	Potential Fix
Small logs missed	Low resolution (256×256)	Train at 512×512
Rock/Ground confusion	Similar texture	Add texture augmentation
Edge imprecision	Downsampling artifacts	Use higher resolution
Shadow misclassification	Color variation	Brightness augmentation

---

5. Optimizations Applied

5.1 Test-Time Augmentation (TTA)

Implemented TTA for inference-time accuracy boost:

- Horizontal flip
- Vertical flip
- Average predictions from 3 views

Expected Improvement: +2% IoU with zero retraining.

5.2 Data Pipeline Optimization

Optimization	Impact
Offline preprocessing	95% I/O reduction
GPU migration	50× speedup
Efficient DataLoader	2× throughput

---

6. Conclusion & Future Work

Key Findings:

1. **Simple architectures win**: Custom UNet outperformed pretrained ResNet34
2. **Class weighting is critical**: Enables detection of rare classes
3. **Domain matters**: ImageNet pretraining doesn't help with synthetic desert data
4. **Preprocessing pays off**: Offline data preparation is worth the investment

Future Improvements:

- [ ] Higher resolution training (512×512) with mixed precision
- [ ] Focal Loss for hard example mining
- [ ] Ensemble of multiple models
- [ ] CRF post-processing for boundary refinement
- [ ] Domain-specific augmentation (desert shadows, dust effects)

---

7. Reproducibility Instructions

Environment Setup:

```
pip install torch torchvision tqdm pillow numpy
```

Training:

```
python train.py
```

Inference:

```
python test.py
```

Files Included:

- `train.py` - Training script with class weighting
- `test.py` - Inference script with TTA
- `checkpoints/best_model.pth` - Trained model weights
- `README.md` - Quick start guide

Thank you for reviewing our submission!

Team Leader: Karthik Rajarapu
Team Members: S. Soheb, Binita, Divya