DesertVision++

Robust Semantic Segmentation for Off-Road Domain Generalization

Team: XDBoost

---

1. Executive Summary

Autonomous off-road navigation requires reliable scene understanding across varying terrain types and environmental conditions. In this project, we developed a deep learning model to perform **pixel-level semantic segmentation** on desert and off-road environments.

Our model achieved:

- **79.65% Mean IoU on validation data**
- **30.41% Mean IoU on unseen test data**

While validation performance was strong, a significant drop on test data revealed a **domain generalization challenge**, which became a central focus of our analysis.

Rather than treating this as a failure, we systematically investigated the domain shift and propose structured improvements for robust real-world deployment.

---

2. Problem Overview (Non-Technical Explanation)

Imagine giving a robot a camera and asking it:

> "Tell me exactly what each pixel in this image represents."

The robot must distinguish:

- Trees
- Bushes
- Dry grass
- Rocks
- Sky
- Logs
- Flowers

- Landscape

This process is called **semantic segmentation**.

The difficulty increases when:

- Lighting changes
- Vegetation density varies
- Texture patterns shift
- Some classes appear very rarely

The real challenge is not performing well on familiar images — it is performing well on **unseen environments**.

---

3. Dataset & Setup

Number of Classes: 10

Classes included: Trees, Lush Bushes, Dry Grass, Dry Bushes, Ground Clutter, Flowers, Logs, Rocks, Landscape, Sky.

Data Split:

- Training set (synthetic)
- Validation set (same distribution)
- Test set (distribution-shifted)

Images were resized to:

```
512 × 512
```

---

4. Model Architecture

We selected:

U-Net++ with ResNet-50 Backbone

Why?

- U-Net++ improves feature fusion across scales
- ResNet-50 provides strong pretrained ImageNet features
- Suitable for small object segmentation (logs, flowers)

Encoder:

ResNet50 (ImageNet pretrained)

Decoder:

Nested skip connections (U-Net++)

Number of Output Classes:

10

---

5. Training Configuration

Loss Function (Hybrid Strategy)

We combined:

- Dice Loss → Optimizes overlap
- Focal Loss → Handles hard-to-classify pixels

Final Loss:

```
0.5 × Dice + 0.5 × Focal
```

This combination helps with:

- Class imbalance
- Small object learning

Optimizer

AdamW Learning Rate: 2e-4 Scheduler: Cosine Annealing (T_max=25)

Batch Size:

4

Training Time:

~2–3 hours (GPU)

---

6. Data Augmentation Strategy

Applied during training:

- Horizontal Flip
- Random Brightness/Contrast
- Hue/Saturation adjustment
- Normalization

These augmentations help simulate:

- Lighting variation
- Mild color changes

However, no geometric distortion or heavy domain randomization was applied.

---

7. Validation Results (In-Distribution)

Best Validation Performance:

Metric	Score
Mean IoU	**0.7965**
Final Val Loss	0.2130

This indicates strong learning capacity on synthetic distribution.

The training curve shows:

- Smooth convergence
- Stable validation behavior
- No severe overfitting signals within synthetic domain

---

8. Test Results (Out-of-Distribution)

Final Mean IoU:

0.3041

Per-Class IoU:

Class	IoU
Trees	0.4003
Lush Bushes	0.0005

Class	IoU
Dry Grass	0.4734
Dry Bushes	0.4438
Ground Clutter	0.0000
Flowers	0.0000
Logs	0.0000
Rocks	0.0497
Landscape	0.6872
Sky	0.9863

---

9. Domain Gap Analysis

The model performs extremely well on:

- Sky (0.9863)
- Landscape (0.6872)

But collapses on:

- Flowers
- Logs
- Ground Clutter
- Lush Bushes

Why?

1️⃣ Class Imbalance

Rare classes appear infrequently in training.

2️⃣ Synthetic Overfitting

Validation data comes from same synthetic distribution.

3️⃣ Micro IoU Bias

Validation used micro reduction, favoring dominant classes.

4️⃣ Texture Sensitivity

Small objects differ significantly between domains.

---

10. Failure Case Observations

Observed issues include:

- Logs misclassified as ground clutter
- Flowers confused with dry grass
- Lush bushes merged with trees
- Rocks mistaken for landscape

This indicates:

- Feature-level confusion
- Insufficient high-frequency robustness

---

11. Strengths of Our Approach

- Strong architectural choice (U-Net++)
- Hybrid loss strategy
- Stable training
- Clean implementation
- Proper metric calculation
- Reproducible pipeline

---

12. Proposed Improvements

To improve domain generalization:

A. Stronger Augmentation

- Gaussian noise
- Motion blur
- Random cropping
- Perspective distortion
- Weather simulation

B. Class Rebalancing

- Weighted Dice loss
- Oversampling rare classes

C. Domain Adaptation

- Feature alignment strategies
- Self-training on pseudo-labels
- Synthetic-to-real adaptation

D. Multi-Scale Training

Improve small object recognition.

---

13. Reproducibility

To train:

```
python train.py
```

To evaluate:

```
python test.py
```

Model weights saved as:

```
offroad_segmentation_model.pth
```

Hardware: GPU acceleration Training time: ~2–3 hours

---

14. Conclusion

DesertVision++ demonstrates:

- Strong performance within known distributions
- Clear exposure of domain generalization limitations
- Structured failure analysis
- Actionable roadmap for improvement

The major insight from this project is:

> High validation accuracy does not guarantee real-world robustness.

Understanding and addressing domain shift is essential for reliable off-road perception systems.