KrackHack 3.0 – AI/ML Track
Off-road Semantic Scene Segmentation
Final Report

1. Team Details

Team Name: HyperBool

Project Repository: Public Github Repository

Model Link: Google Drive Link

For Reproducibility: README.md

Members

- **Aryan Sisodiya (Team Leader)** – InfinityXR9
- **Daksh Rathi** – DakshRathi-India
- **Farhan Alam** – Frozen-AFK
- **Tanmay Pratap Singh** – DhmalTPS

2. Executive Summary

Goal. Build a multiclass semantic segmentation model for autonomous off-road navigation in desert environments using synthetic training data, while maximizing robustness under domain shift (novel but visually similar desert scenes).

Outcome. We developed the system through five phases driven by failure-case analysis: (1) a DeepLabV3+ CNN baseline that reached **0.58** validation mIoU but ignored rare obstacles, (2) a stronger DeepLabV3+ CNN with a ResNet-50 backbone and aggressive domain-randomizing augmentation that reached **0.64** validation mIoU but overfit, (3) DeepLabV3+ CNN stabilization using Stochastic Weight Averaging (SWA) activated at the **20th epoch** to reduce overfitting (with a lower but more generalized mIoU), (4) a SegFormer Transformer shift that achieved **0.67** validation mIoU with **0.43** mIoU on the blind test set (Train Loss **0.59**, Validation Loss **0.50**), and (5) final optimization focused on reducing the remaining validation-to-test domain gap and packaging the final training/inference workflow for submission.

3. Problem Statement & Core Challenges

Autonomous navigation in desert environments requires accurate pixel-level recognition of terrain, vegetation, obstacles, and sky. Training on synthetic digital twins is attractive, but introduces two dominant challenges:

(1) Synthetic-to-target domain gap. Models exploit texture shortcuts (stable sand color, stylized shadows, clean edges) instead of learning geometry and physical structure, causing strong validation scores but weak robustness on shifted test distributions.

(2) Extreme class imbalance. Large "easy" classes dominate pixel counts, while safety-relevant classes (e.g., clutter, rocks, logs, sparse vegetation) occupy a very small fraction of

pixels. Without explicit imbalance control, the model optimizes overall mIoU by ignoring rare objects.

These constraints required deliberate choices beyond a standard segmentation recipe: break texture memorization, increase rare-class contribution to learning, diagnose overfitting, and validate against blind test shift.

4. Dataset Overview

4.1 Structure

```
dataset/
  train/
    Color_Images/
    Segmentation/
  val/
    Color_Images/
    Segmentation/
training_dataset/
    Color_Images/
    Segmentations/
```

4.2 Semantic Classes

The dataset uses integer class IDs. The following mapping is used throughout training and evaluation.

Table 1: Semantic classes (IDs and names).

Class ID	Class Name
100	Trees
200	Lush Bushes
300	Dry Grass
500	Dry Bushes
550	Ground Clutter
600	Flowers
700	Logs
800	Rocks
7100	Landscape (general ground not in other categories)
10000	Sky

4.3 Evaluation Metric

Performance is measured using mean Intersection-over-Union (mIoU). For class c:

$$IoU_c = \frac{TP_c}{TP_c + FP_c + FN_c}, \qquad mIoU = \frac{1}{C}\sum_{c=1}^{C} IoU_c$$

where TP, FP, FN denote pixel-level true positives, false positives, and false negatives, and C is the number of classes.

5. Methodology: Phased Development

We adopted an iterative, failure-mode-driven approach. Each phase was judged on metrics and on qualitative behavior under domain shift.

5.1 Phase 1: Baseline Prototype (DeepLabV3+ CNN & EfficientNet-B0) — Val mIoU = 0.58

Baseline configuration.

- Architecture: DeepLabV3+ CNN
- Encoder: EfficientNet-B0 (pretrained)
- Input resolution: 256×256
- Loss: Dice Loss + Focal Loss (unweighted)
- Augmentations: minor/basic photometric and geometric transforms

Observed behavior. The baseline achieved **0.58** validation mIoU, but this score was inflated by easy-class dominance:

- the unweighted objective allowed strong prediction of massive "sand/landscape" and "sky" regions
- rare, safety-critical obstacles (e.g., rocks, logs, clutter, thin structures) were ignored or under-detected

This established the core failure pattern: a DeepLabV3+ CNN can appear strong on validation while failing on rare classes.

5.2 Phase 2: Domain Gap Mitigation & Overfitting (DeepLabV3+ CNN & ResNet-50) – Val mIoU = 0.64 (Overfitting Observed)

To correct Phase-1 failure modes, we upgraded the DeepLabV3+ CNN pipeline and aggressively attacked texture shortcuts.

Architecture upgrade.

- DeepLabV3+ CNN backbone upgraded to ResNet-50 to learn deeper feature representations

Bridging the domain gap (aggressive augmentation).

- Strong appearance disruption: ColorJitter, RandomShadow, RandomBrightnessContrast
- Texture destruction: GaussianBlur, Gaussian noise, image compression
- Spatial robustness: Shift/Scale/Rotate and flips

Why this works. Synthetic renders enable DeepLabV3+ CNNs to "cheat" by memorizing stable RGB textures. Aggressive color/shadow/blur/noise destroys these shortcuts and forces learning of structural geometry.

Tackling class imbalance (weighted supervision).

- We computed class imbalance from mask pixel statistics and used it to weight the Dice+Focal supervision so that rare-class misses are penalized far more strongly than easy-class pixels.

Result. These changes improved validation performance to **0.64** mIoU, but the model exhibited **overfitting** to the training distribution.

5.3 Phase 3: Stabilization via Stochastic Weight Averaging (DeepLabV3+ CNN + SWA)

To address the overfitting observed in Phase 2, we introduced stabilization late in training.

SWA activation.

- SWA was activated at the **20th epoch**.
- The learning rate was increased during the SWA stage to move away from sharp, overfitted minima and explore a wider region of the loss landscape.

Weight averaging effect.

- Across the final epochs, the model weights were averaged to form a flatter, more generalizable solution.

Result. Overfitting was reduced. The mIoU decreased compared to Phase 2, but the model behavior was more honest and generalized under shift.

5.4 Phase 4: SegFormer Transformer Paradigm Shift (SegFormer Transformer) – Val mIoU = 0.67, Blind Test mIoU = 0.43

Recognizing the limitations of DeepLabV3+ CNN local receptive fields, we shifted to a SegFormer Transformer to leverage global context.

SegFormer Transformer motivation.

- SegFormer Transformer self-attention captures long-range context, improving global consistency across weak-texture desert scenes.
- Context helps separate visually similar regions using scene structure instead of texture.

Regularization.

- DropPath regularization was used during training.
- Observed losses: Train Loss **0.59**, Validation Loss **0.50**.

Results.

- Validation mIoU = **0.67**
- Blind Test mIoU = **0.43**

This confirmed that the SegFormer Transformer was healthier and stronger on validation, but a significant blind-test domain gap remained.

5.5 Phase 5: Final Optimization & Submission Pipeline

Phase 5 focused on reducing the remaining validation-to-test gap and producing a clean, reproducible final pipeline:

- tighter training stabilization and checkpoint selection for the SegFormer Transformer
- stress-testing on blind `testImages` to diagnose remaining failure cases
- final inference packaging: resizing, normalization, forward pass, argmax decoding, remapping to original class IDs, and saving masks in the submission format

6. Experimental Setup

6.1 Protocol (No Test Leakage)

Training used only `train/` data and validation used only `val/` data for selection and monitoring. The `training_dataset/` folder was used only for blind evaluation and robustness checks.

6.2 Implementation and Compute

Software: Python, PyTorch, segmentation-models-pytorch, Albumentations, OpenCV, Jupyter.

Hardware: NVIDIA RTX 4060 (8GB), mixed precision enabled.

6.3 Key Hyperparameters (Used in Phases)

- Phase 1 input size: 256×256; batch size: 8; epochs: 30
- Phase 2 input size: 512×512; batch size: 4; epochs: 30
- Optimizer: Adam (Phase 1 DeepLabV3+ CNN), AdamW (Phase 2 DeepLabV3+ CNN with differential learning rates)
- Scheduler: cosine annealing schedule in Phase 2 DeepLabV3+ CNN
- Loss: Dice Loss + Focal Loss (unweighted in Phase 1; weighted supervision introduced from Phase 2)

7. Results and Analysis

7.1 Quantitative Summary

7.2 Per-class Behavior

Include class-wise IoU to show rare-class improvements and remaining failure modes.

Table 2: Model progression (phase-wise).

Model / Phase	**Architecture**	**Key Outcome**
Phase 1 Baseline	DeepLabV3+ CNN (EffNet-B0)	Val mIoU = 0.58; rare-class collapse
Phase 2 Domain Gap + Imbalance	DeepLabV3+ CNN (ResNet-50)	Val mIoU = 0.64; overfitting observed
Phase 3 Stabilization	DeepLabV3+ CNN + SWA (epoch 20+)	Overfitting reduced; mIoU decreased but generalized
Phase 4 SegFormer Transformer	SegFormer Transformer	Val mIoU = 0.67; Blind Test mIoU = 0.43
Phase 5 Final Pipeline	SegFormer Transformer (optimized)	Domain-gap reduction + final submission packaging

Table 3: Class-wise IoU on validation (fill with measured values).

Class ID	**Class**	**IoU (Train)**	**IoU (Test)**
100	Trees	0.8652	0.4093
200	Lush Bushes	0.7133	0.0009
300	Dry Grass	0.7070	0.4721
500	Dry Bushes	0.5155	0.4693
550	Ground Clutter	0.3938	NOT PRESENT in GT
600	Flowers	0.6799	NOT PRESENT in GT
700	Logs	0.6251	NOT PRESENT in GT
800	Rocks	0.5372	0.0520
7100	Landscape	0.6971	0.6806
10000	Sky	0.9849	0.9821
Mean		**0.6719**	**0.4380**

7.3 Learning Curves

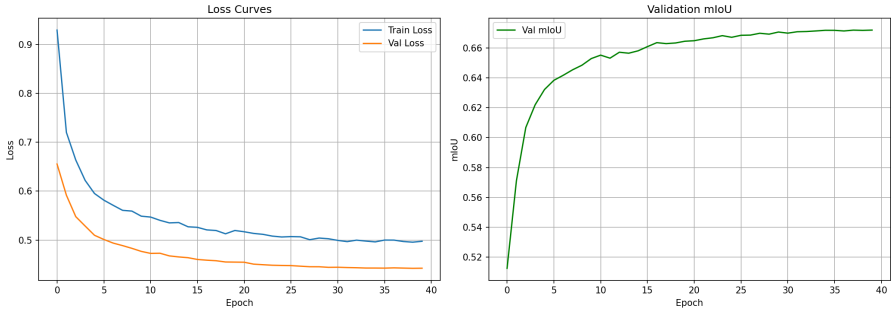
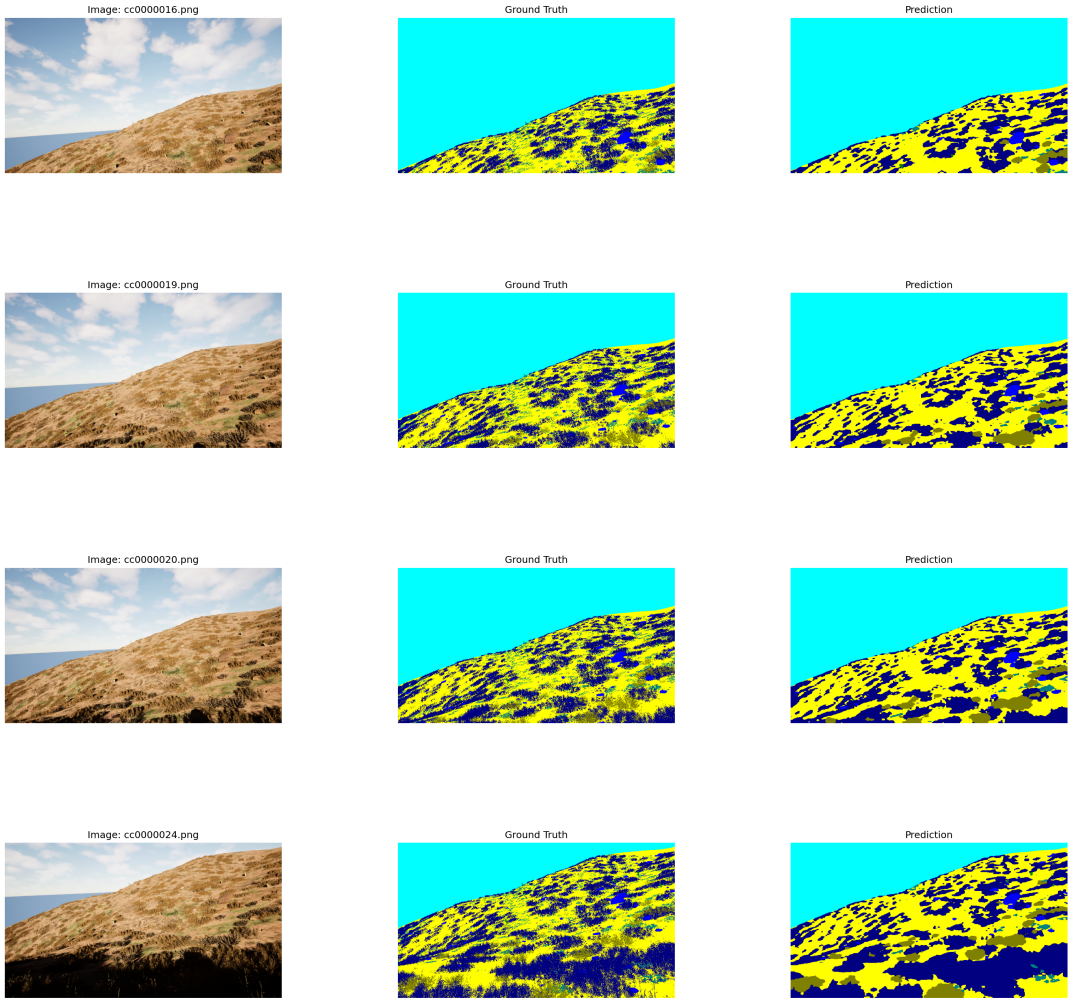


Figure 1: Training vs validation loss and mIoU curves across phases. (Insert plotted curves.)

7.4 Qualitative Results and Failure Cases

Observed improvements.

- Phase 2 DeepLabV3+ CNN improved robustness to lighting/texture via RandomShadow, ColorJitter, blur/noise



(a) RGB (b) Ground Truth (c) Prediction

Figure 2: Qualitative examples across easy and hard cases

- Phase 3 DeepLabV3+ CNN + SWA reduced overfitting behavior seen in Phase 2
- Phase 4 SegFormer Transformer improved global consistency and stability across desert scenes

Remaining failure modes.

- visually similar classes: logs vs rocks; clutter vs dry bush
- small objects far from the camera (thin structures, distant obstacles)
- strong validation-to-test domain gap on blind test distribution

8. Inference Pipeline

Inference followed a deterministic pipeline aligned with training preprocessing:

- load RGB images from `testImages/`
- resize to the trained input resolution and apply the same normalization
- forward pass through the selected checkpoint

- decode predictions via argmax over classes
- remap contiguous class indices back to original mask IDs (100–10000)
- save predicted masks in the required submission folder structure

9. Ablations

We report controlled comparisons that reflect the actual phase progression:

- DeepLabV3+ CNN (EfficientNet-B0) vs DeepLabV3+ CNN (ResNet-50)
- minor/basic augmentation vs aggressive domain-randomizing augmentation
- DeepLabV3+ CNN (ResNet-50) vs DeepLabV3+ CNN (ResNet-50 + SWA)
- DeepLabV3+ CNN best run vs SegFormer Transformer

Table 4: Ablation template (fill with measured values).

Setting	**Val mIoU**	**Notes**
DeepLabV3+ CNN (EffNet-B0, Phase 1)	0.58	unweighted Dice+Focal; rare-class collapse
DeepLabV3+ CNN (ResNet-50, Phase 2)	0.64	aggressive augmentation; overfitting observed
DeepLabV3+ CNN (ResNet-50 + SWA, Phase 3)	—	overfitting reduced; mIoU lower but generalized
SegFormer Transformer (Phase 4)	0.67	Blind Test mIoU = 0.43
SegFormer Transformer (Phase 5)	—	optimized to reduce domain gap

10. Current Status & Future Work

Current best model. SegFormer Transformer achieved **0.67** validation mIoU and **0.43** blind test mIoU, with Train Loss **0.59** and Validation Loss **0.50**. Phase 5 targets the remaining domain gap.

Next steps.

- reduce validation-to-test domain gap through targeted domain randomization and harder stress tests
- strengthen rare-class learning using focused sampling on obstacle-heavy patches
- add boundary-focused supervision to reduce edge bleeding between similar terrain categories

11. Conclusion

This project followed a failure-mode-driven engineering process: establish a DeepLabV3+ CNN baseline, diagnose rare-class collapse and synthetic texture shortcuts, attack the domain gap

with strong augmentation and imbalance-aware supervision, reduce Phase-2 overfitting using SWA activated at the 20th epoch, and shift to a SegFormer Transformer to leverage global context. The final challenge is shrinking the blind-test domain gap under true distribution shift, addressed in Phase 5 through final optimization and clean submission packaging.