03_Test_Images_and_Scripts_Insights

03 — Test Images & Scripts Insights

Folder Location

```
DATASET/
├── Offroad_Segmentation_testImages/
│   └── Offroad_Segmentation_testImages/
│       ├── Color_Images/    (342 PNG files)
│       └── Segmentation/    (342 PNG files)
│
└── Offroad_Segmentation_Scripts/
    ├── train_segmentation.py
    ├── test_segmentation.py
    ├── visualize.py
    └── ENV_SETUP/
        ├── setup_env.bat
        ├── create_env.bat
        └── install_packages.bat
```

Test Images

Property	Details
Total Images	**342** Color + **342** Segmentation masks
Image Format	`.png`
Naming Convention	`0000060.png` to `0000XXX.png` — **different prefix** from train/val (no `cc` prefix)
Nested Folder	Note the double nesting: `Offroad_Segmentation_testImages/Offroad_Segmentation_testImages`
Has Segmentation Masks	**YES** — contrary to the hackathon doc which says "testImages (RGB only, no masks)"

> **Surprise**: The hackathon PDF says test images should be RGB-only with no masks. But your download includes segmentation masks for test images too! This is either a bonus for local evaluation, or the organizers included them by accident. Either way — **NEVER train on test data** (instant disqualification).

Test Dataset vs Train/Val Comparison

Aspect	Train	Val	Test
Count	293	289	342
Filename Prefix	`cc`	`cc`	None (just numbers)
Has Masks	Yes	Yes	Yes (unexpected)
Domain	Desert Scene A	Desert Scene A (diff location)	Desert Scene B (**different desert**)
Purpose	Learn patterns	Monitor overfitting	Final benchmark on unseen locale

Total dataset: 924 images across all three splits.

Scripts Breakdown

`train_segmentation.py` (591 lines) — The Main Training Script

Config	Value	Notes
Backbone	`DINOv2 ViT-Small/14` (`dinov2_vits14`)	Pre-trained, frozen (no gradients)
Seg Head	`SegmentationHeadConvNeXt`	Conv stem → depthwise conv block → classifier
Loss	`CrossEntropyLoss`	Standard for multi-class segmentation
Optimizer	`SGD(lr=1e-4, momentum=0.9)`	Conservative LR
Batch Size	`2`	Small — needed for 6GB VRAM
Epochs	`10`	Very few — likely needs 30-50+
Image Size	`476 × 266` (W × H)	Derived from `960/2` and `540/2`, snapped to multiples of 14
Classes	`10` (including background)	But only 9 annotated + 1 background
Augmentations	**NONE**	Critical gap — must add augmentations
Scheduler	**NONE**	No learning rate scheduling

Config	Value	Notes
Best Model Save	**NO**	Only saves final epoch
Output	`train_stats/` folder	Loss curves, IoU curves, Dice curves, metrics text file

Architecture Flow:

```
Input Image → DINOv2 (frozen) → Patch Tokens → ConvNeXt Head → Per-pixel
logits → Bilinear Upsample → CrossEntropy Loss
```

Available backbone sizes (in the code but only "small" is used):

- `small` = `vits14` (21M params) ← current
- `base` = `vitb14_reg` (86M params)
- `large` = `vitl14_reg` (300M params)
- `giant` = `vitg14_reg` (1.1B params) — won't fit in 6GB VRAM

`test_segmentation.py` (488 lines) — Inference & Evaluation Script

Feature	Details
Purpose	Load trained weights → run on test/val images → save predictions + metrics
CLI Arguments	`--model_path` , `--data_dir` , `--output_dir` , `--batch_size` , `--num_samples`
Default Model	`segmentation_head.pth` (in scripts folder)
Default Data	Points to `Offroad_Segmentation_testImages/`
Outputs	`masks/` (raw class IDs), `masks_color/` (RGB visualization), `comparisons/` (side-by-side), metrics summary
Color Palette	10 defined colors for visualization (black, forest green, lime, tan, brown, olive, saddle brown, gray, sienna, sky blue)

Color Palette for Classes:

Class	Color
Background	Black `[0,0,0]`
Trees	Forest Green `[34,139,34]`

Lush Bushes	Lime `[0,255,0]`
Dry Grass	Tan `[210,180,140]`
Dry Bushes	Brown `[139,90,43]`
Ground Clutter	Olive `[128,128,0]`
Logs	Saddle Brown `[139,69,19]`
Rocks	Gray `[128,128,128]`
Landscape	Sienna `[160,82,45]`
Sky	Sky Blue `[135,206,235]`

`visualize.py` (54 lines) — Quick Mask Colorizer

- Takes a folder of segmentation masks → assigns random colors to unique pixel values → saves colorized PNGs
- **Input folder is blank** (`" "`) — you need to set it manually
- Useful for quickly eyeballing what the masks look like

ENV_SETUP Scripts (Windows .bat files)

File	What It Does
`setup_env.bat`	Runs `create_env.bat` then `install_packages.bat` in sequence
`create_env.bat`	`conda create --name EDU python=3.10 -y`
`install_packages.bat`	`conda install pytorch torchvision pytorch-cuda=11.8 ultralytics -y && pip install opencv-contrib-python && pip install tqdm`

> **Note**: These scripts use **Conda** and create an `EDU` environment. Since we already set up a **venv**, we'll install the packages directly into our venv instead, skipping these bat files entirely.

Critical Improvement Opportunities

1. **Add data augmentations** — #1 priority (random flip, rotation, color jitter, random crop)
2. **Increase epochs** — 10 is way too few; try 30–50

3. **Add learning rate scheduler** — `CosineAnnealingLR` or `StepLR`
4. **Save best model** — track `best_val_iou` and save checkpoint
5. **Try larger backbone** — `vitb14_reg` (base) if VRAM allows
6. **Add early stopping** — prevent wasting time if model plateaus
7. **Consider mixed precision** — `torch.cuda.amp` for faster training on RTX 3050
8. **Add class weights to loss** — handle class imbalance (Landscape/Sky dominate)