01_Training_Dataset_Insights

01 — Training Dataset Insights

Folder Location

```
DATASET/Offroad_Segmentation_Training_Dataset/train/
├── Color_Images/    (293 PNG files)
└── Segmentation/    (293 PNG files)
```

What's Inside

Property	Details
Total Image Pairs	**293** (RGB + mask)
Image Format	`.png`
Naming Convention	`cc0000012.png` to `cc0000XXX.png` (prefixed with `cc`, not perfectly sequential — some IDs are skipped)
Color Images	Standard RGB desert scene photos (synthetic, from Falcon/Duality AI simulator)
Segmentation Masks	Grayscale PNGs where each pixel value = class ID
Image-Mask Pairing	**Same filename** in both folders (e.g., `Color_Images/cc0000012.png` ↔ `Segmentation/cc0000012.png`)

Pixel-Level Class Mapping (Segmentation Masks)

The masks use these **raw pixel values** (not 0–9 directly):

Raw Pixel Value	Class ID (Mapped)	Class Name	Description
0	0	Background	Empty / unlabeled
100	1	Trees	Cacti, desert trees
200	2	Lush Bushes	Greener bushes
300	3	Dry Grass	Dry/brown grass patches
500	4	Dry Bushes	Sparser, drier bushes

Raw Pixel Value	Class ID (Mapped)	Class Name	Description
550	5	Ground Clutter	Small debris, rocks, trash
700	6	Logs	Fallen wood, branches
800	7	Rocks	Stones, boulders
7100	8	Landscape	General traversable ground
10000	9	Sky	Sky / background

> **Important**: The `convert_mask()` function in `train_segmentation.py` remaps these raw values to 0–9 during loading. You never work with the raw values directly in training — the DataLoader handles it.

How This Data is Loaded (from `train_segmentation.py`)

Python

```python
class MaskDataset(Dataset):
    # Reads from Color_Images/ and Segmentation/ subfolders
    # Pairs images by SAME filename
    # Applies convert_mask() to remap pixel values → class IDs (0-9)
    # Returns (image_tensor, mask_tensor)
```

Transforms applied during training:

- `Resize((266, 476))` — width: `int(((960/2) // 14) * 14) = 476`, height: `int(((540/2) // 14) * 14) = 266`
- `ToTensor()` — converts to float [0, 1]
- `Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])` — ImageNet normalization
- Masks: `Resize` + `ToTensor` then multiplied by 255 to get back to integer class IDs

---

Key Observations

1. **293 images is a small dataset** — you'll need augmentations (flip, rotate, color jitter, etc.) to prevent overfitting

2. **Synthetic data** — all images are computer-generated desert scenes from Duality Falcon, not real photos
3. **10 classes total** — including background and sky, so 8 "real" object classes
4. **Class imbalance is expected** — Landscape and Sky likely dominate most frames; Flowers (class 600) is listed in the hackathon doc but **NOT in the code's value_map** (important!)
5. **No Flowers class in code** — The hackathon PDF mentions class ID 600 (Flowers) but the provided `train_segmentation.py` doesn't include it. Either the dataset doesn't contain flowers, or they're treated as background
6. **Image resolution** — original synthetic images are likely 960×540 (based on the resize formula), which gets halved and snapped to multiples of 14 for DINOv2 patch tokens

---

Potential Issues to Watch

- **Missing IDs in filenames** — numbering has gaps (e.g., `cc0000032` exists but some nearby numbers don't). This is normal but means you can't iterate by number
- **Small batch size** — default `batch_size=2` is set because DINOv2 backbone is memory-heavy on a 6GB GPU. This is appropriate for your RTX 3050
- **No built-in augmentations** — the provided script has zero augmentations — adding them is the #1 way to boost performance