PyTorch Implementation of UNet for Land Cover Segmentation with Partial Cross-Entropy Loss

1. Introduction

This project implements semantic segmentation under **partial supervision** — training models with only a fraction of pixels labeled. We compare UNet and UNet++ architectures on land cover segmentation using a custom Partial Cross-Entropy loss function.

---

2. Method

2.1 Partial Cross-Entropy Loss

A custom loss function designed for partial supervision scenarios where only some pixels have ground-truth labels.

Mathematical Formulation:

$$L_{partial} = -\frac{1}{|V|} \sum_{i \in V} \log \text{softmax}_{y_i}(z_i)$$

where:

- V = set of pixels with known labels (not marked as `ignore_index = -1`)
- $|V|$ = number of valid labeled pixels
- z_i = model logits at pixel i
- y_i = ground truth class at pixel i

Key Properties:

- Computes loss **only** on labeled pixels
- Automatically ignores unlabeled pixels (`ignore_index = -1`)
- Normalizes by number of labeled pixels (not total pixels)
- Enables training with sparse annotations

2.2 Model Architectures

Both architectures implemented from scratch in pure PyTorch:

UNet: 5-level encoder-decoder with skip connections, 64 base channels, BatchNorm + ReLU

UNet++: Nested decoder with dense skip connections

Configuration: Input 384×384 RGB images, output 5-class segmentation (Background, Buildings, Woodlands, Water, Roads)

---

3. Experimental Setup

3.1 Dataset

LandCover.ai: 41 orthophotos (9636×9095 pixels), split into 512×512 patches (resized to 384×384)

- Training: 7,470 patches
- Validation: 1,602 patches
- Classes: 5 (background, buildings, woodlands, water, roads)
- Augmentation: Horizontal/vertical flip, rotation, shift-scale-rotate

3.2 Training Configuration

Parameter	Value
Optimizer	Adam
Learning Rate	1e-5
Weight Decay	1e-4
Batch Size	32
Epochs	12
Mixed Precision	Yes (AMP)
Device	CUDA
Random Seed	42

3.3 Experiments

4 experiments: 2 architectures (UNet, UNet++) × 2 label fractions (10%, 15%). All evaluated on fully labeled validation set.

---

3. Results

Architecture	Label Fraction	Best mIoU	Final mIoU	Final Accuracy
UNet++	10%	0.5054	0.5054	0.8680
UNet	15%	0.4902	0.4764	0.8605
UNet	10%	0.4887	0.4775	0.8640
UNet++	15%	0.4705	0.4630	0.8398

Observations:

- UNet++ @ 10% achieved best performance (0.5054 mIoU)
- UNet++ @ 15% performed worse than @ 10%
- UNet showed consistent improvement from 10% to 15%
- Peak performance reached by epoch 10-12

Class Weights: `[0.66, 1.27, 1.12, 1.03, 1.91]` applied to all experiments

---

4. Summary

Results:

- Partial CE loss trained models with 10-15% labeled pixels
- Best: UNet++ @ 10% (0.5054 mIoU, 86.80% accuracy)
- UNet: 2x faster training than UNet++ (1.5 vs 3 min/epoch)
- Label increase 10%→15%: UNet improved +0.3%, UNet++ decreased -6.9%

Limitations:

- Limited label fractions tested (10%, 15%)
- Random pixel masking only
- Single dataset, 12 epochs, validation-only evaluation

---

5. Resources

If you want the trained models and the ready-to-use cropped data, I uploaded both on this link:

- **Google Drive Link**

Appendix

Environment:

- PyTorch 2 with CUDA support
- Mixed Precision Training (AMP)
- Random seed: 42
- Hardware: A100 80GB vRAM (Azure Virtual Machine)

Training Time: UNet ~18 min, UNet++ ~36 min (12 epochs)

Saved Models: `runs/{unet,unetplusplus}_frac{10,15}/best_model.pth`

Implementation: From-scratch PyTorch | **Date:** November 2025