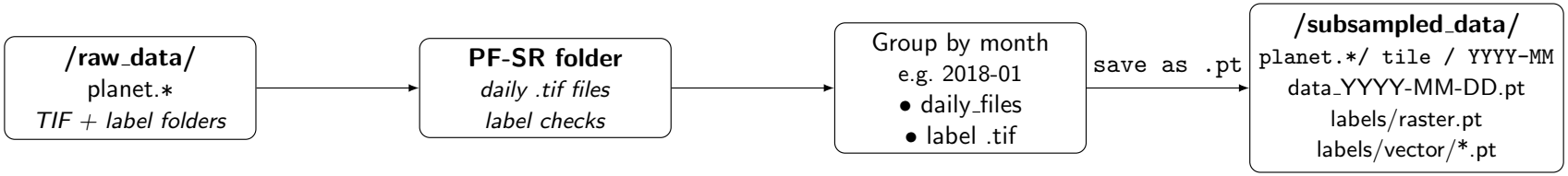
Pipeline description and first results

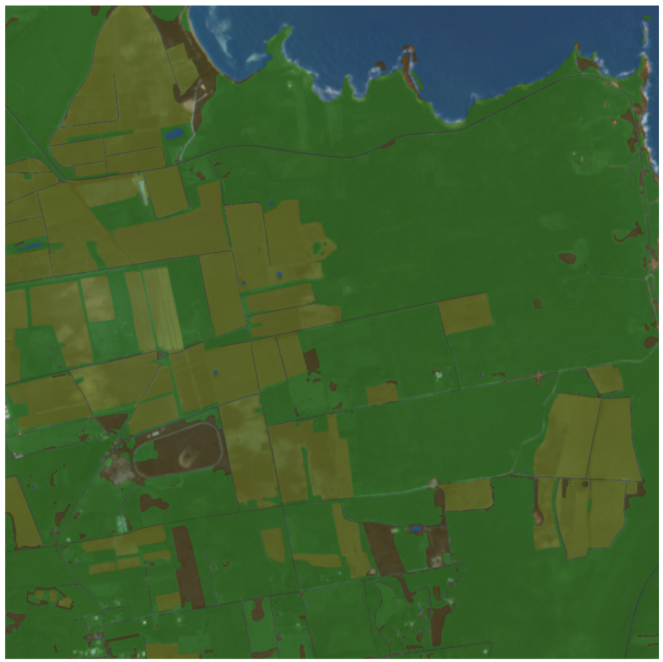
Table of Contents

load_data_subsampled.py



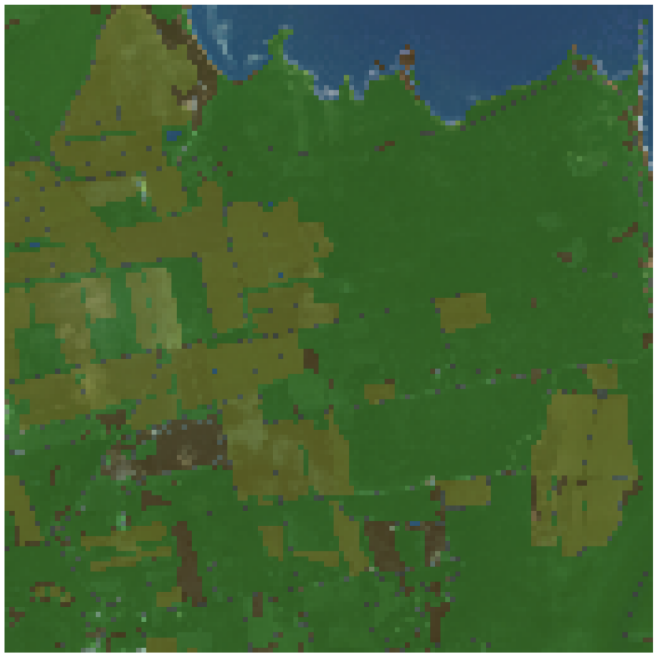
Subsample every 8 pixels.

2018-01



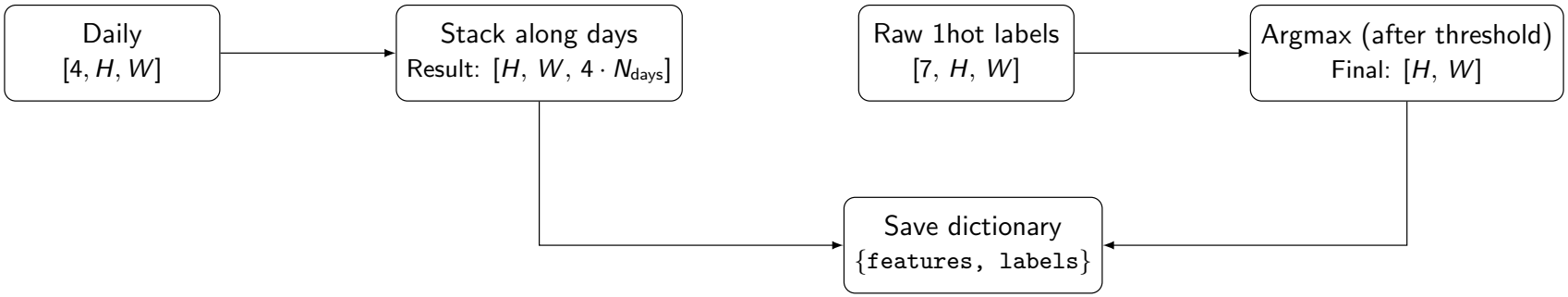
1024×1024

2018-01



128×128

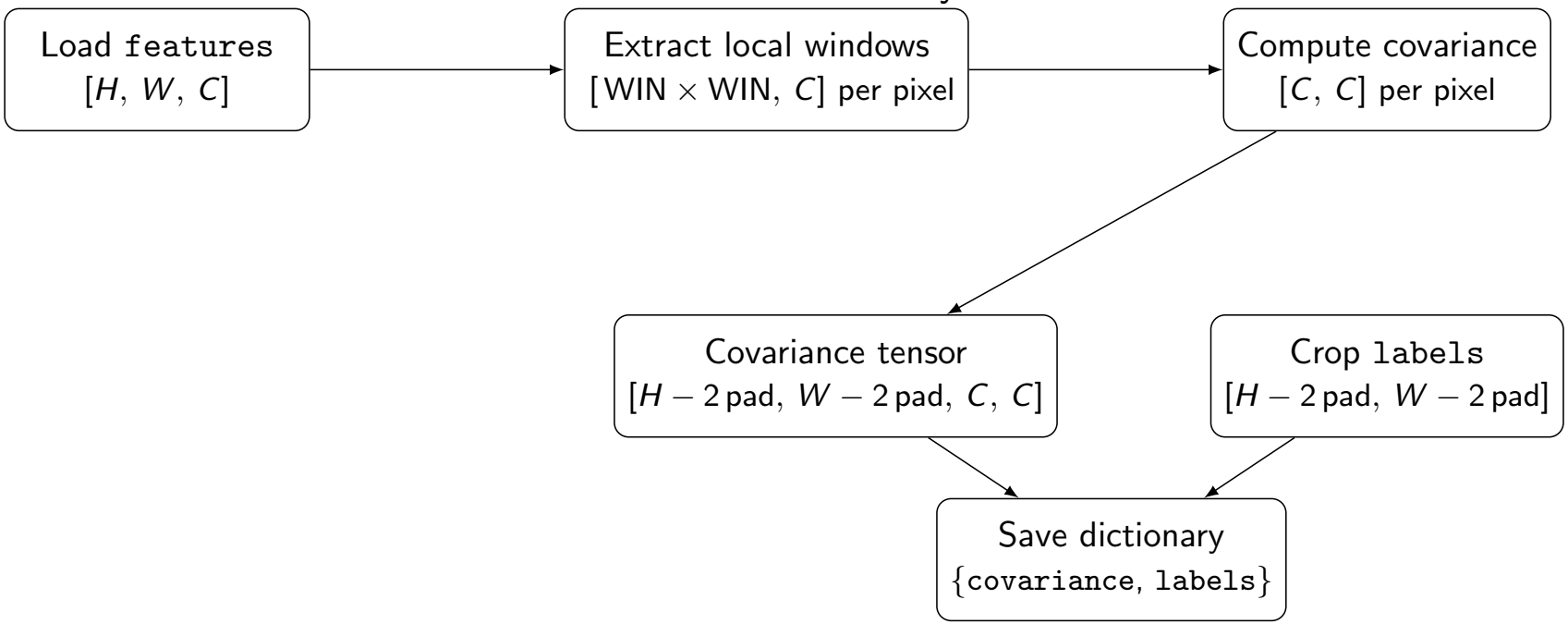
month_stacked_label.py




```
subsampled_data
└── datasets
    └── unet
        └── planet.*
            └── tile_id
                └── YYYY-MM
                    └── pixel_dataset_YYYY-MM.pt
```

cov_label.py

$H = W = 128$, WIN $= 11$, $C = 4 \times N_{\text{days}}$.



- Covariance: [118,118,4×N_{days}], Labels [118,118]
- SPD-ness safeguards: noise=1e-6, spd_eps=1e-5, alpha=1e-4

SPD-ness safeguards

$$x_i' = x_i - \bar{x} + \text{NOISE} \times \mathcal{N}(0, id)$$

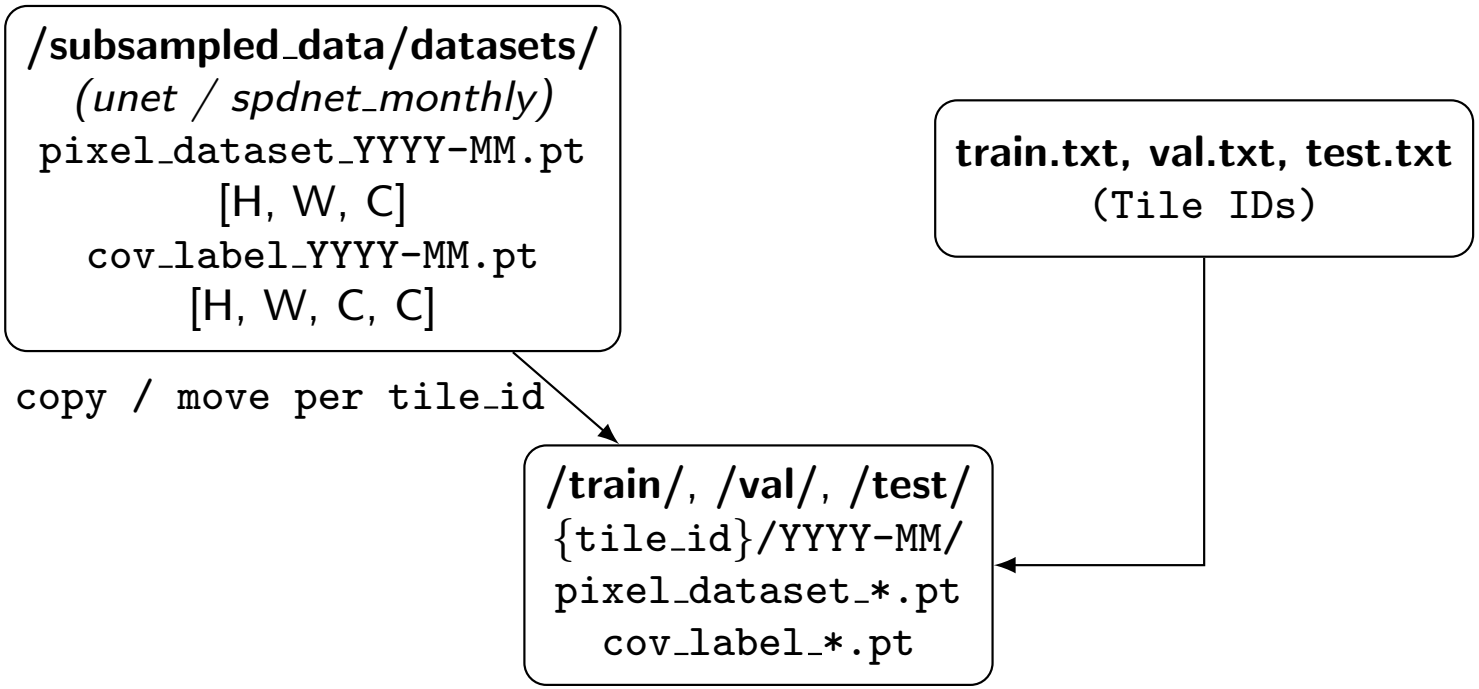
$$\Sigma = \frac{1}{n-1} \sum_{i=1}^{n} x_i' (x_i')^T$$

$$\Sigma_{reg} = \Sigma + \alpha I$$

$$\Sigma_{sym} = 0.5(\Sigma_{reg} + \Sigma_{reg}^T) = Q \text{diag}(\lambda_i) Q^T$$

$$\boxed{\Sigma_{SPD} = Q \text{diag}(\max(\lambda_i, \text{spd}_{eps})) Q^T}$$

reorg_train_val_test.py



Samples from train/val/test

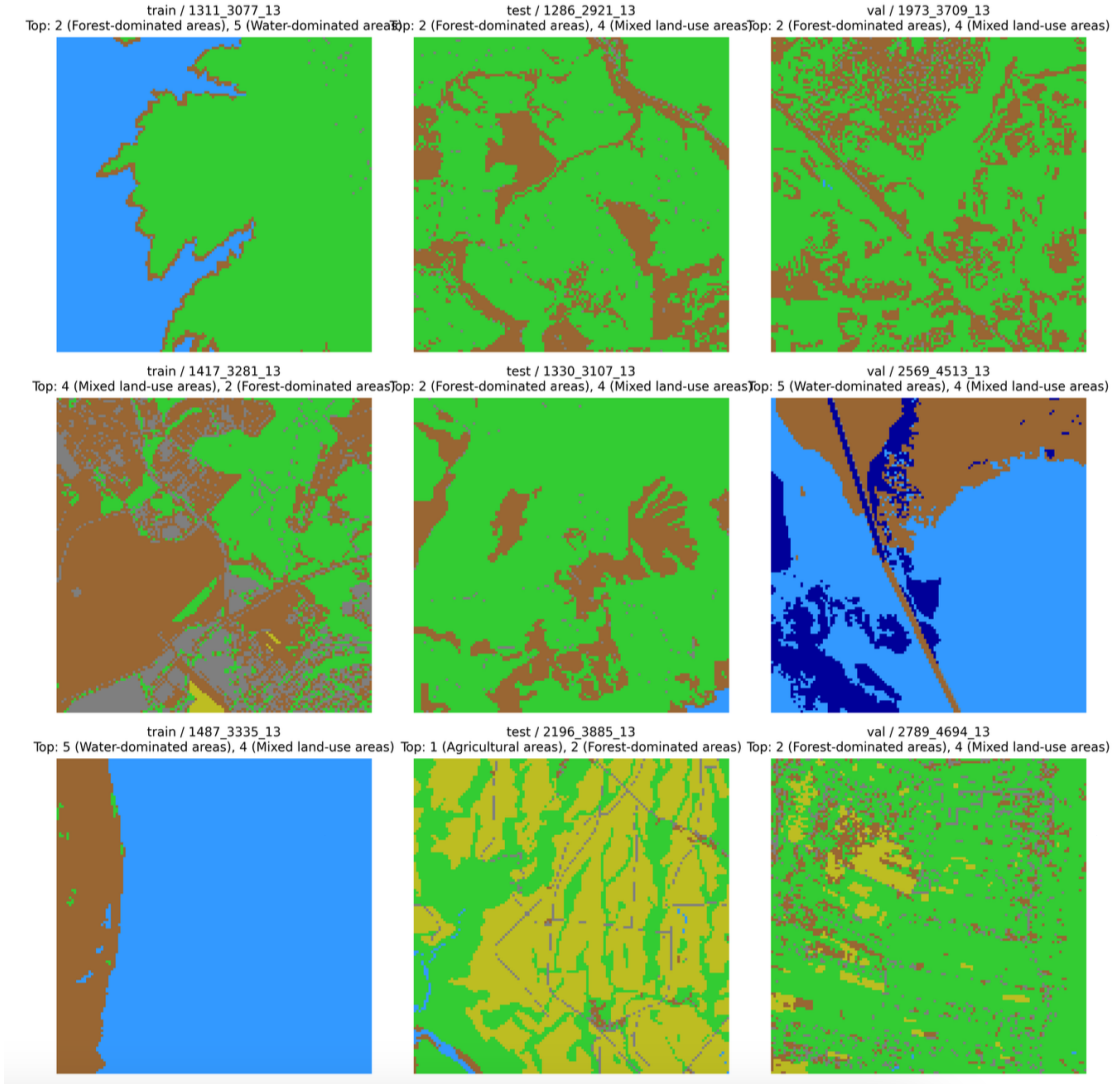
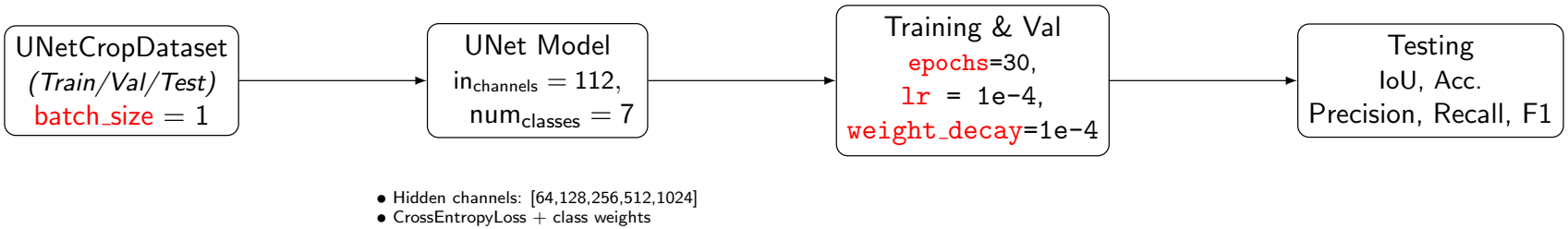


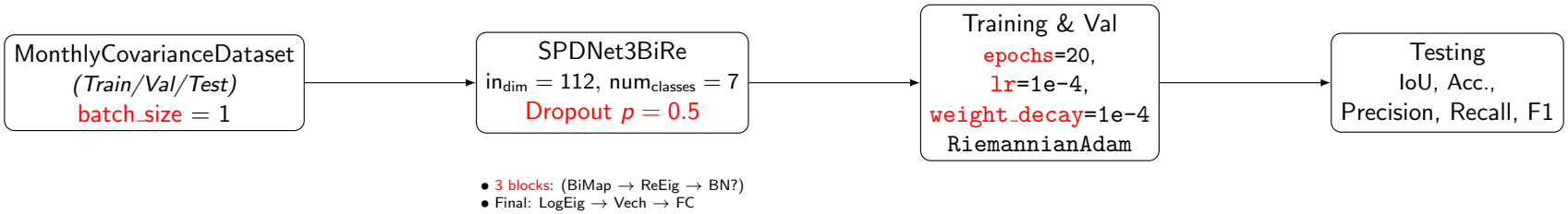
Figure: In practice: 16 tiles/3 tiles/3 tiles

unet_pipeline.py



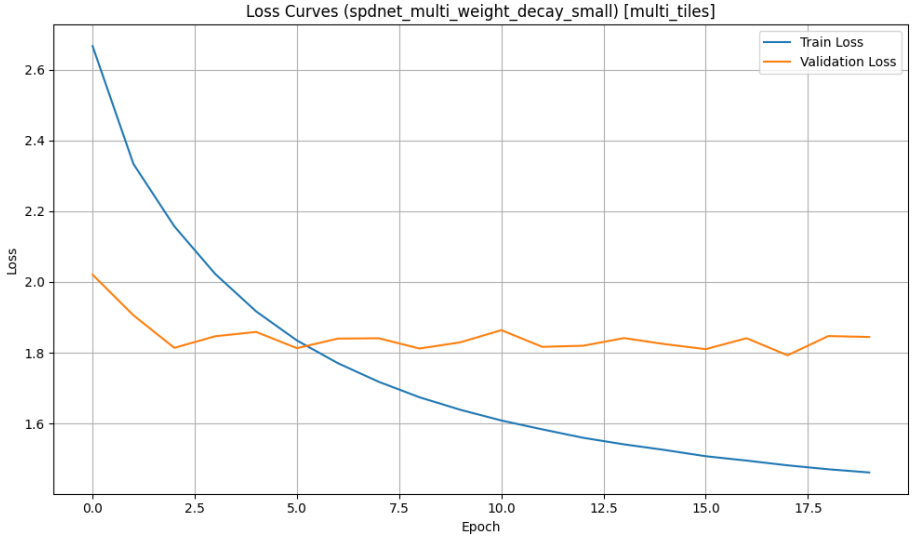
- Seed
- Dataset: features $[118 \times 118, 4 \times 28]$, labels $[118 \times 118]$
- Training with class weighted CE, weight decay for Adam.

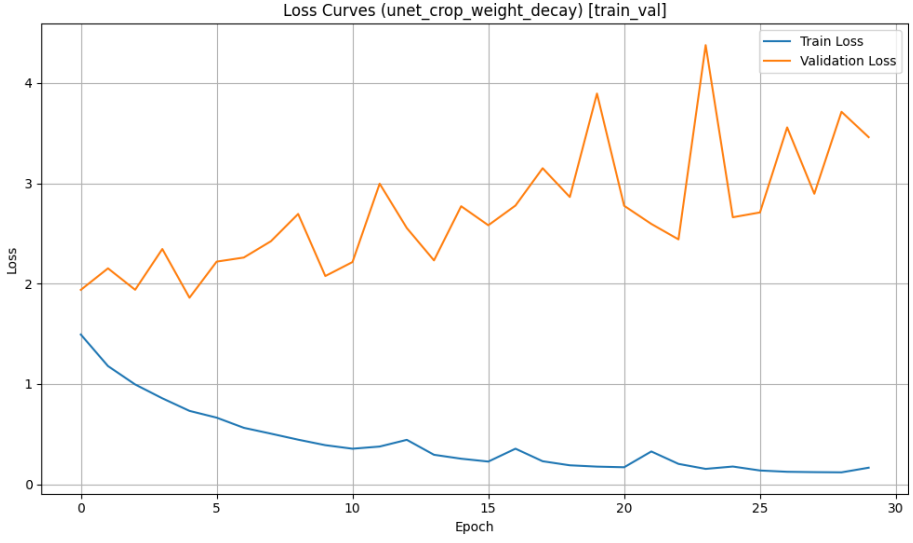
basic_spdnet_pipeline.py



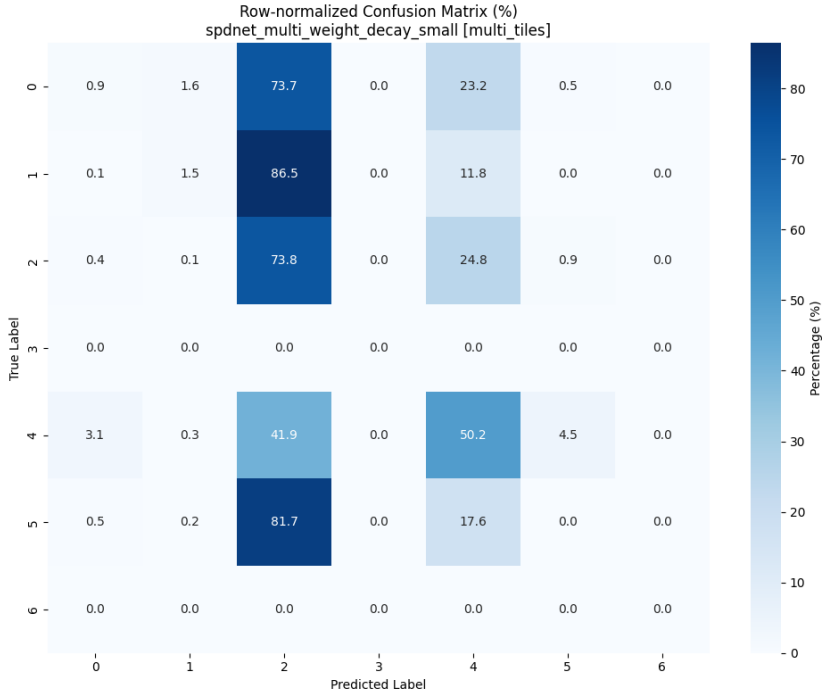
- Seed
- Defining SPDNet3BiRe $(118 \to 64 \to 32 \to 16)$, ensuring spd-ness.
- ReEIG ε, clamp parameter.
- Dataset class: covariance $[118 \times 118, 4 \times 28, 4 \times 28]$, labels $[118 \times 118]$.
- Training with class weighted CE, weight decay for Riemannian Adam optimizer.

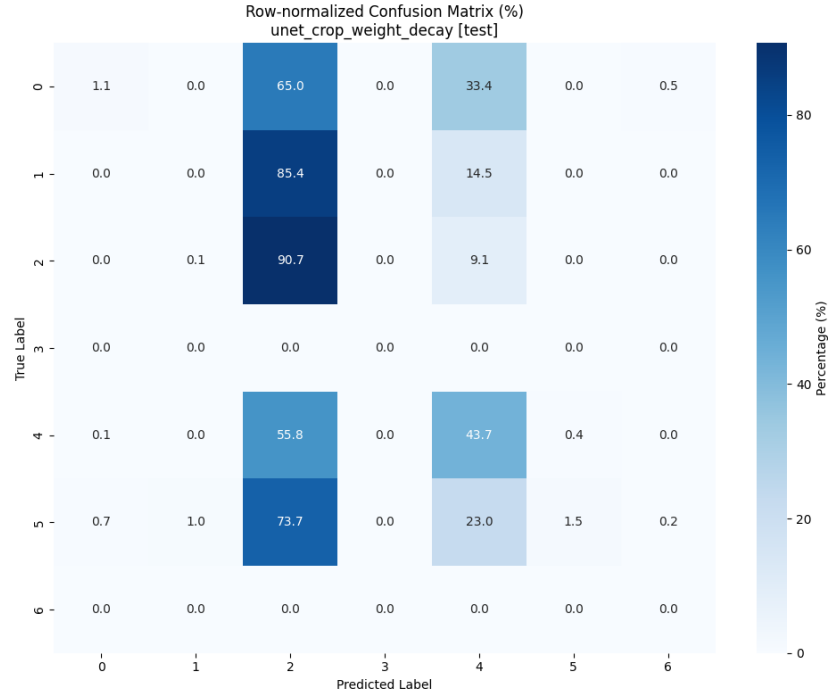
Overfitting!





Per-class accuracy

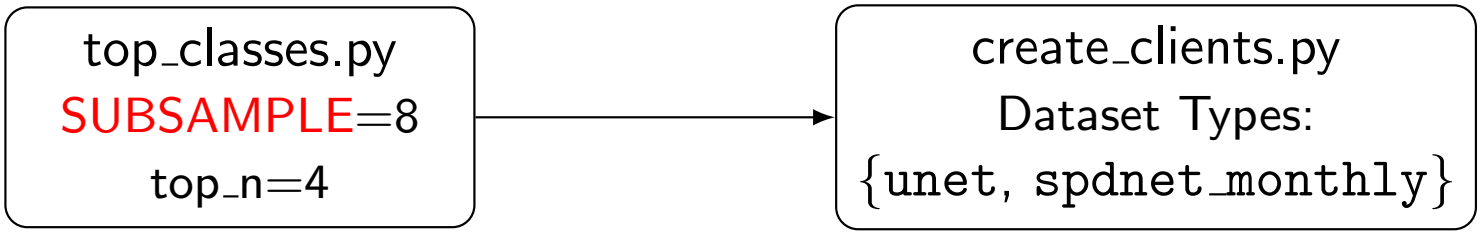




Test Accuracy	0.5408
Test mIoU	0.1290
Precision (macro)	0.2778
Recall (macro)	0.2107
F1 Score (macro)	0.1837

Test Accuracy	0.6355
Test mIoU	0.1329
Precision (macro)	0.2142
Recall (macro)	0.1956
F1 Score (macro)	0.1786

Data grouping & client creation



- Collect per-tile label frequencies
- Group tiles by dominant land class
- Copy tile data into client folders

urban/	Urban areas	0 (imp. surfaces)
agri/	Agricultural areas	1 (agriculture)
forest/	Forest-dominated areas	2 (forest etc.)
mixed/	Mixed land-use areas	3 (wetlands), 4 (soil), 6 (snow/ice)
water/	Water-dominated areas	5 (water)

Clientwise training

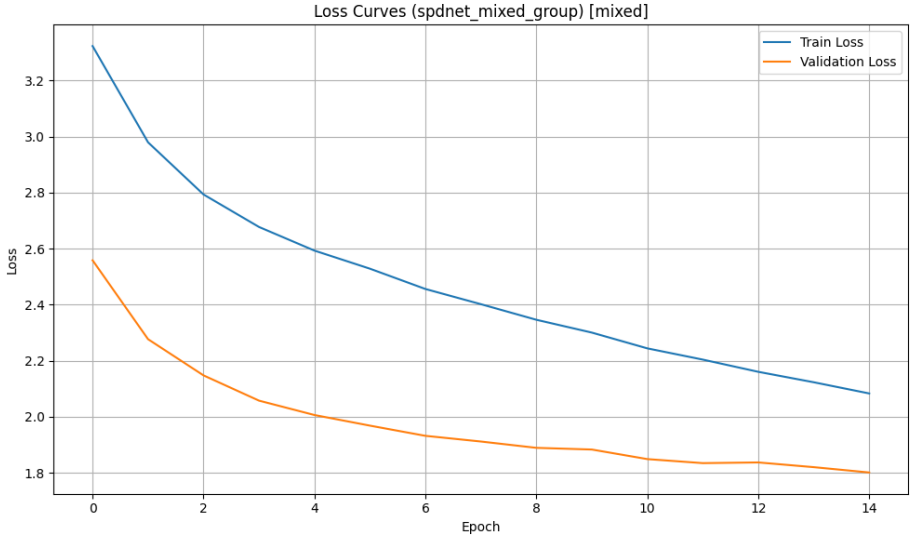
Instead of varied data (as in train/, val/, test/), train on homogeneous data.

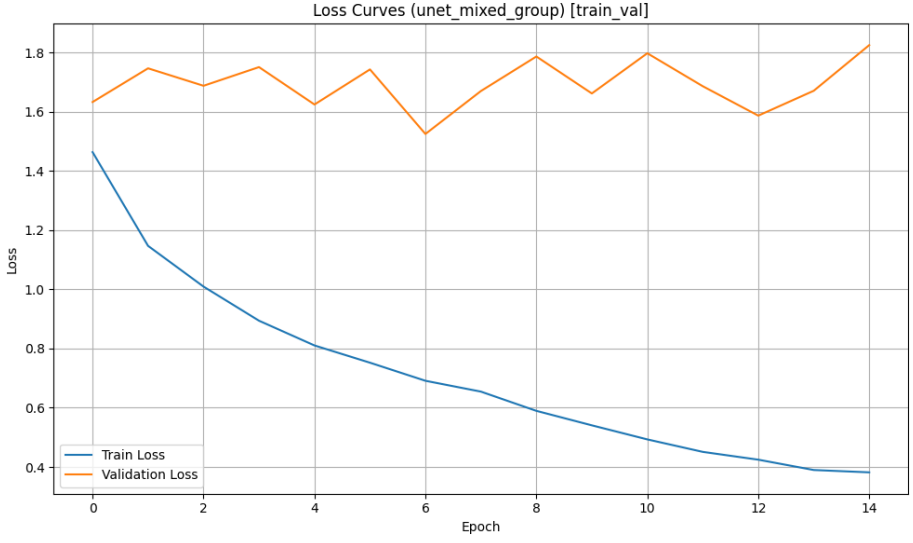
unet_clientwise.py
T=28 → in_channels=112
epochs=15, lr=1e-4,
weight_decay=1e-4
train:val:test=70:15:15
Adam

spdnet_clientwise.py
T=28 → input_dim=112
epochs=15, lr=1e-4
weight_decay=1e-4
train:val:test=70:15:15
RiemannianAdam

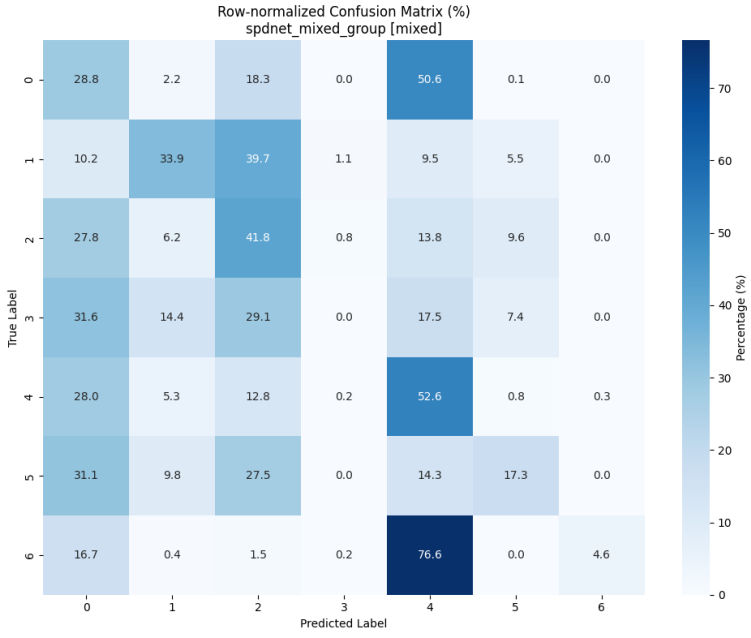
- mixed/ (5 tiles)
- water/ (3 tiles)

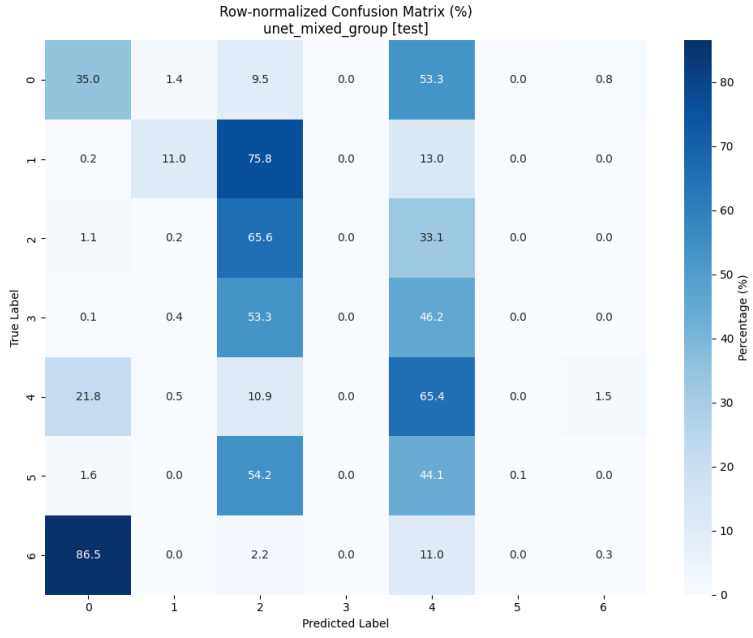
mixed/


Loss Curves (spdnet_mixed_group) [mixed]
Train Loss
Validation Loss
Loss
Epoch
3.2
3.0
2.8
2.6
2.4
2.2
2.0
1.8
0
2
4
6
8
10
12
14


Loss Curves (unet_mixed_group) [train_val]
Train Loss
Validation Loss
Loss
Epoch
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0
2
4
6
8
10
12
14

mixed/

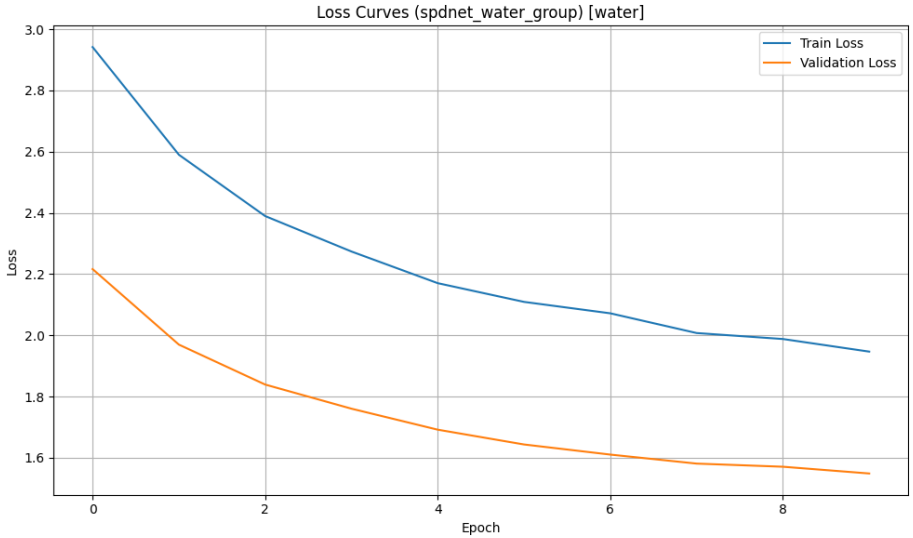


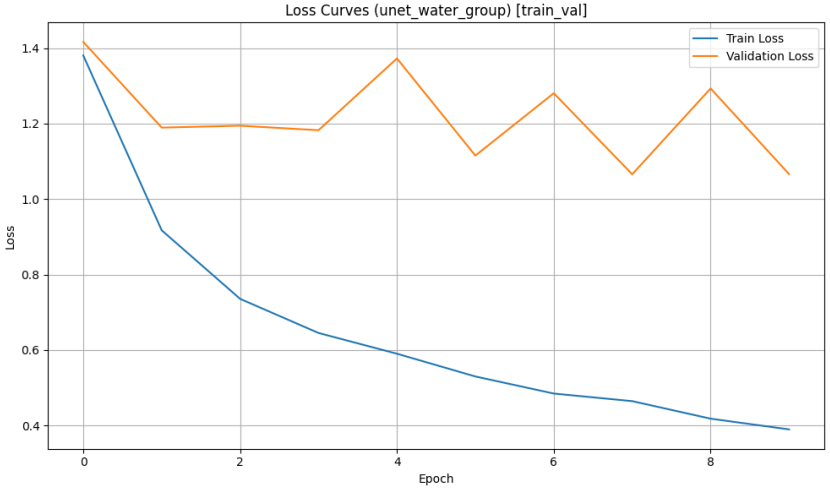


Test Accuracy	0.4369
Test mIoU	0.1491
Precision (macro)	0.3296
Recall (macro)	0.2558
F1 Score (macro)	0.2319

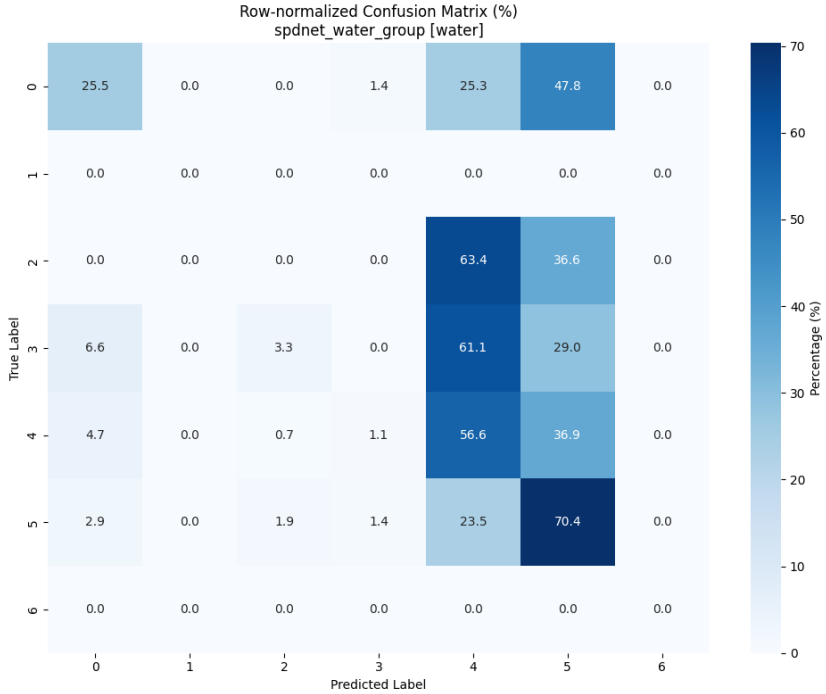
Test Accuracy	0.4251
Test mIoU	0.1447
Precision (macro)	0.4374
Recall (macro)	0.2535
F1 Score (macro)	0.2127

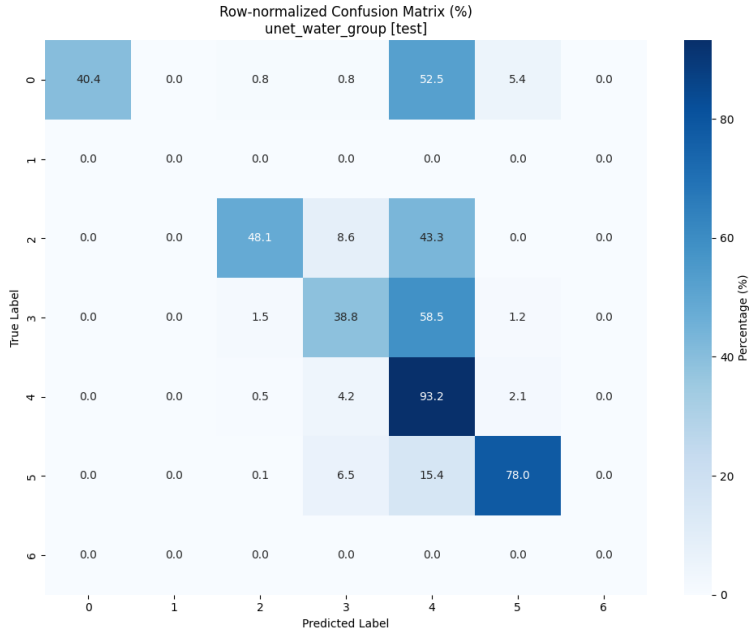
water/


Loss Curves (spdnet_water_group) [water]
Train Loss
Validation Loss
Loss
Epoch
3.0
2.8
2.6
2.4
2.2
2.0
1.8
1.6
0
2
4
6
8


Loss Curves (unet_water_group) [train_val]
Train Loss
Validation Loss
Loss
Epoch
1.4
1.2
1.0
0.8
0.6
0.4
0
2
4
6
8

water/





Test Accuracy	0.6467
Test mIoU	0.1885
Precision (macro)	0.2514
Recall (macro)	0.3050
F1 Score (macro)	0.2529

Test Accuracy	0.7946
Test mIoU	0.4478
Precision (macro)	0.6472
Recall (macro)	0.5970
F1 Score (macro)	0.5877

Possible improvements

Main pb: u-net overfits

- Decrease the size of u-net
- Data augmentation
- Higher batch_size
- LR Scheduler (plateau)
- Decrease/Increase nb of spd-layers
- Choice of loss function w.r.t. imbalance dataset
- Early stopping
- Increase nb of epochs
- Hyperparameter tuning