

Python-based deep learning for detecting ditches from elevation data

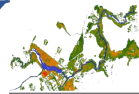
Holger Virro & Wai Tik Chan

Nordic-RSE conference

May 31, 2024



UNIVERSITY
OF TARTU



Landscape Geoinformatics Lab

Where we come from

Home institution: Department of
Geography, **University of Tartu**

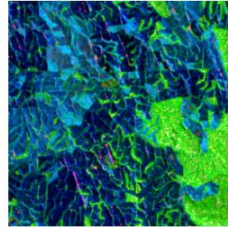
Our workgroup: **Landscape
Geoinformatics Lab**, led by **Prof
Evelyn Uuemaa**

<https://landscape-geoinformatics.ut.ee/>

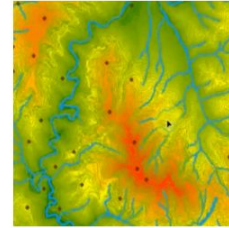
<https://github.com/LandscapeGeoinformatics>



Spatial Analysis and
Spatial Machine Learning



Environmental Remote
Sensing



Environmental Modelling

Who we are

Holger Virro

- Research Fellow in Geoinformatics
- PhD in Geoinformatics, University of Tartu

Wai Tik Chan

- Scientific Programmer
- MSc in Computer Science, University of Tartu



My perspective

From a researcher point of view

- No experience in writing “production” software
- No previous experience with DL
- What should I handle on my own?
- What should I let the RSE handle?

General background

Estonian Land Board (Maa-amet)

- Manages Estonian geospatial datasets
- Interested in leveraging AI to make mapping more efficient
- Inspired by Finnish National Land Survey



Three-dimensional spatial data of the Land Board received an award

Project background

Why **ditches**?

- Estonia has a dense drainage ditch network
- Many ditches not mapped as they are hard to detect from orthophotos
- Accurate ditch data required for modeling hydrology, carbon fluxes etc.

Perhaps **deep learning** (DL) can be used to map them?

Starting point

Found an existing ditch detection model from Sweden (**Lidberg et al., 2023**)

- **U-Net** architecture
- Input: digital elevation model (DEM) images processed with high-pass median filter (**HPMF**)
- **TensorFlow 2.6**
- Packaged in a **Docker** image
- <https://doi.org/10.1061/JIDEDH.IRENG-9796>

Issues with trying to run existing model

No experience with Docker

- Took some much time to set it up

Still did not run properly in the end

- “Graphics card and GPU difference” issue according to author 😞

General vision for the project

Develop a DL workflow for the team

- Should be based on **PyTorch** to align with specific geospatial packages (TorchGeo, Xarray)
- Allows to test different parameter **configurations** (kernel size, batch size etc)
- Data processing pipeline (image sliding, cloud etc.)
- Can be repurposed for other projects
- Can be used on our local machine as well as **HPC**

Workflow components

1. **U-Net class** for the model architecture
 - Should handle different kernel sizes
2. **Data loader** for reading and preprocessing images
 - Optional data augmentation
3. **DL “runner”** to glue components together, train and test the model
 - Should handle loading existing weights for finetuning purposes
4. **Configuration** file to serve as runner input
 - Specifies kernel, batch size, padding, epochs etc.

Researcher



Config.yaml

Data

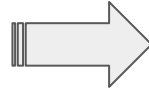
- Location (cloud, HPC)
- pre-processing

Model

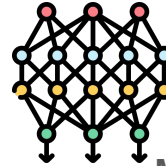
- Architecture
- Parameters

Experiment

- Metrics
- output



RSE



Model



- Experiments Result
- Application

Config file example

```
1 experimentname: &experimentname 'unet_ksize_3x3_relu_sweden_unaug'
2 experimentpath: &experimentpath './experiments'
3 model:
4   unet.UNET:
5     pooling: 'max'
6     in_channel: 1
7     hiddenactivation: 'ReLU'
8     encoder_ksize: [3, 3]
9     decoder_ksize: [3, 3]
10    encoder_padding: 1
11    decoder_padding: 1
12  loss:
13    BCEWithLogitsLoss:
14    #MSELoss:
15    #HuberLoss:
16  #weight:
17  opt:
18    Adam:
19      lr: 0.0001
20  epoch: 100
21  dataset: 'TrainTestDataSet'
22  datasetloader:
23    batch_size: 16
24    shuffle: True
25  accuracymetric: # sklearn.metrics
26    threshold: 0.5
27    BinaryF1Score:
28    # you can add parameter for the function if need
29    BinaryAccuracy:
30    BinaryPrecision:
31    BinaryRecall:
32  weightexport:
33    path: [ *experimentpath, *experimentname, 'weight' ]
34    selection: 'best' # support best or all
35    additional_criterion: 'BinaryF1Score'
36  experimentresult:
37    path: [ *experimentpath, *experimentname, 'result' ]
38    exportSample: False
39  data:
40    loader: 'SKImgDataLoader'
41    filetype: 'tif'
42  training:
43    data: '../deep_learning/data/pretraining/training/hpmf'
44    label: '../deep_learning/data/pretraining/training/labels'
```

HPC script example

```
1  #!/bin/bash
2
3  #SBATCH -J finetuned_aug_estonia_relu_ks7x7_500ep_bs16_from_train_unaug_sweden_test_unaug_sweden
4  #SBATCH -N 1
5  #SBATCH --cpus-per-task=4
6  #SBATCH -t 72:00:00
7  #SBATCH --mem=8G
8  #SBATCH --partition=gpu
9  #SBATCH --gres=gpu:tesla:1
10 #SBATCH -o /gpfs/space/home/holgerv/Ditches/working/pytorch_unet/log/%j_%x.out
11
12 # Experiment name
13 experiment=finetuned_aug_estonia_relu_ks7x7_500ep_bs16_from_train_unaug_sweden_test_unaug_sweden
14
15 cd ~/Ditches/working/pytorch_unet
16
17 rm -rf ./experiments/$experiment/result
18 rm -rf ./experiments/$experiment/weight
19
20 module load python/3.10.10
21
22 python DLRunner.py -config ./experiments/$experiment/config.yml
```

Thanks for listening!

