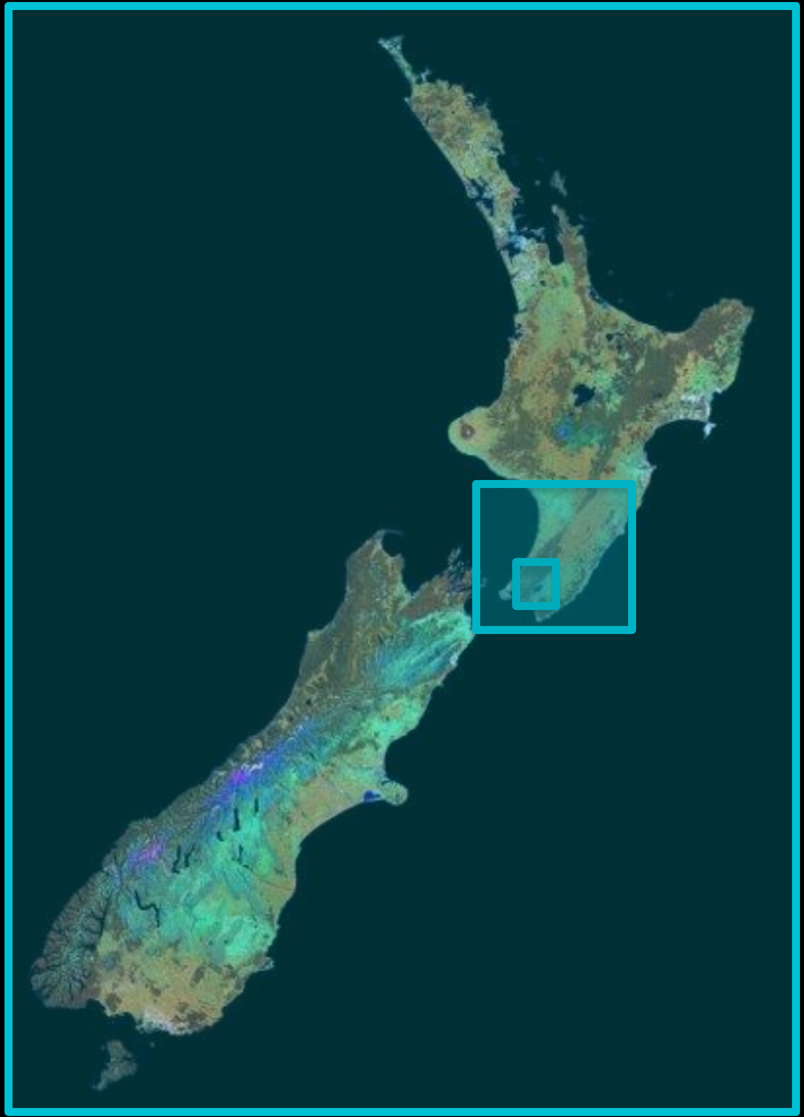
Manaaki Whenua
Landcare Research

Experiences developing an operational workflow for large-scale instance and semantic segmentation of remote sensing imagery using CNNs

Jan Schindler, Brent Martin, Alexander Amies, Ben Jolly and David Pairman

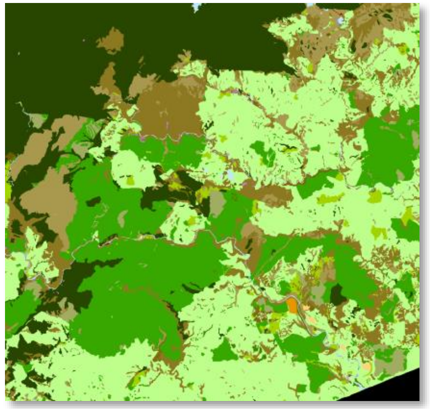
New Zealand Research Software Engineering Conference | 17 September 2021

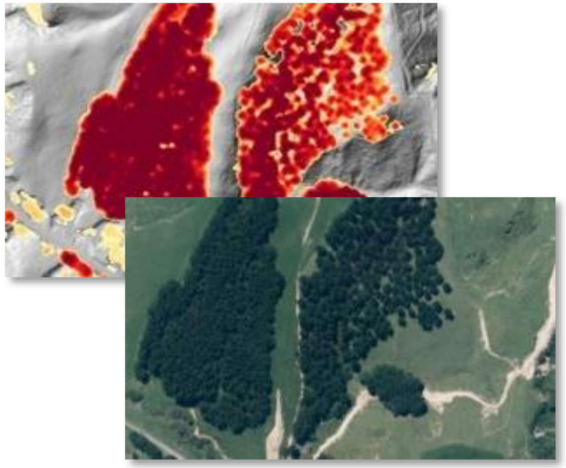
Motivation



Convolutional Neural Networks (CNN) offer great opportunities for improved mapping of land cover, change and individual objects in remote sensing imagery

MWLR hosts a wide range of datasets that are ideal for deep learning tasks:
land cover database, environmental layers, soil maps, Sentinel-1/2 cloud-free imagery archive and seasonal mosaic, LiDAR data, ...





Need for a flexible, operational workflow for mapping exercises at local-, regional- and national scale

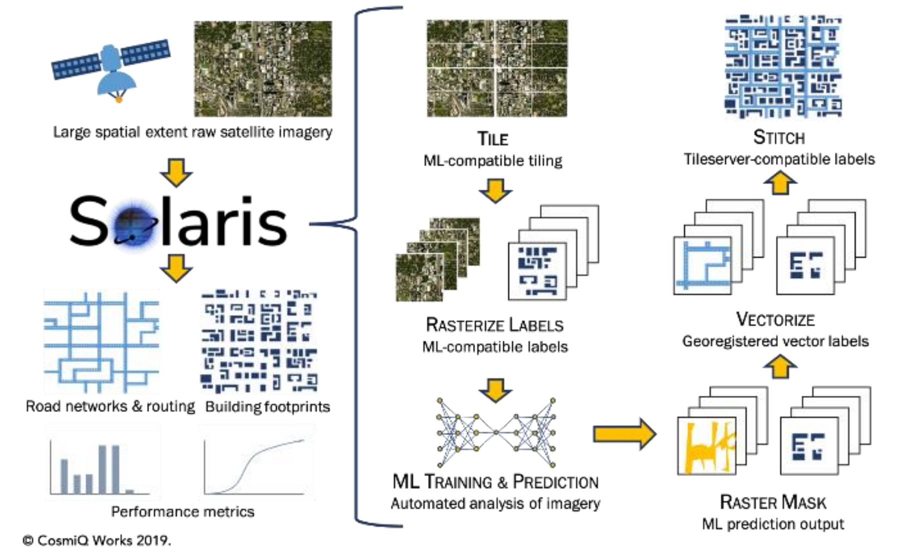
Why not use existing open source DL pipelines?

- Optimized for geospatial imagery
- No need to worry about data processing
- Great tutorials and documentation
- Optimized workflows for classification / detection tasks
- Built-in popular DL models

- Very large code base
- Locked-in to specific package versions
- Cumbersome to set up
- Difficult to extend or customise
- Not ideal for specific data/infrastructure needs
- Problems with NeSI HPC
- Create thousands of tiny files

Solaris

An open source ML pipeline for overhead imagery by CosmiQ Works



© CosmiQ Works 2019.

https://github.com/CosmiQ/solaris

Lots of others, but some abandoned or too specific…

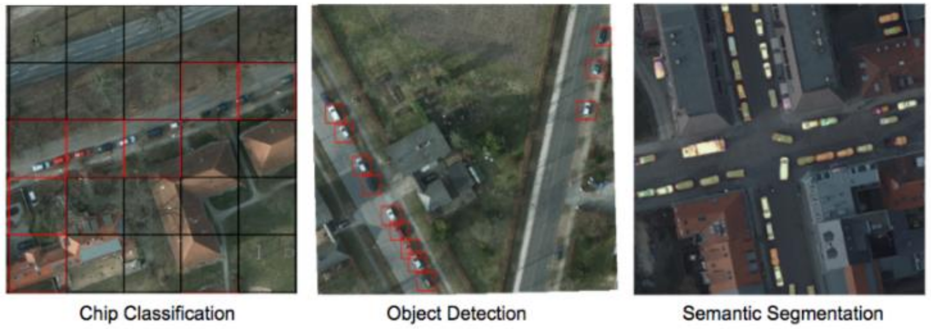
TorchSat

TorchSat is an open-source deep learning framework for satellite imagery analysis based on PyTorch.

https://github.com/sshuair/torchsat

rastervision

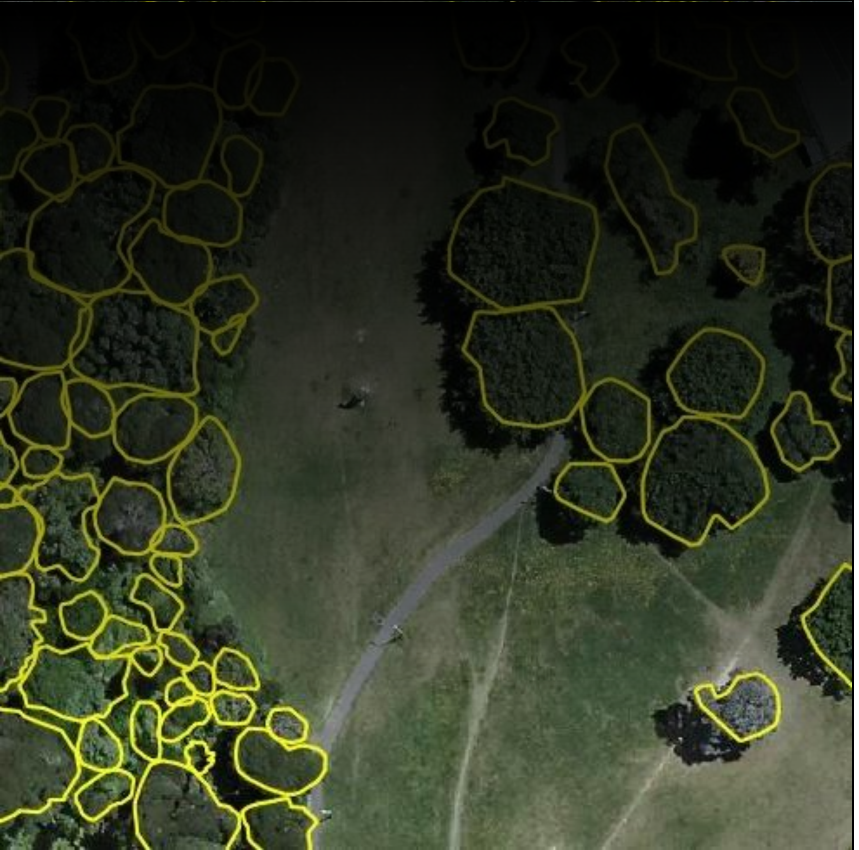
Raster Vision is an open source framework for Python developers building computer vision models on satellite, aerial, and other large imagery sets (including oblique drone imagery). There is built-in support for chip classification, object detection, and semantic segmentation using PyTorch.



https://github.com/azavea/raster-vision

DeepSeg

Semantic and instance segmentation for remote sensing imagery



We created a fit-for-purpose, reusable software pipeline that:

- runs on the NeSI HPC and on local PCs (with GPU support)
- allows for flexible training from geospatial layers and large volumes of remote sensing imagery

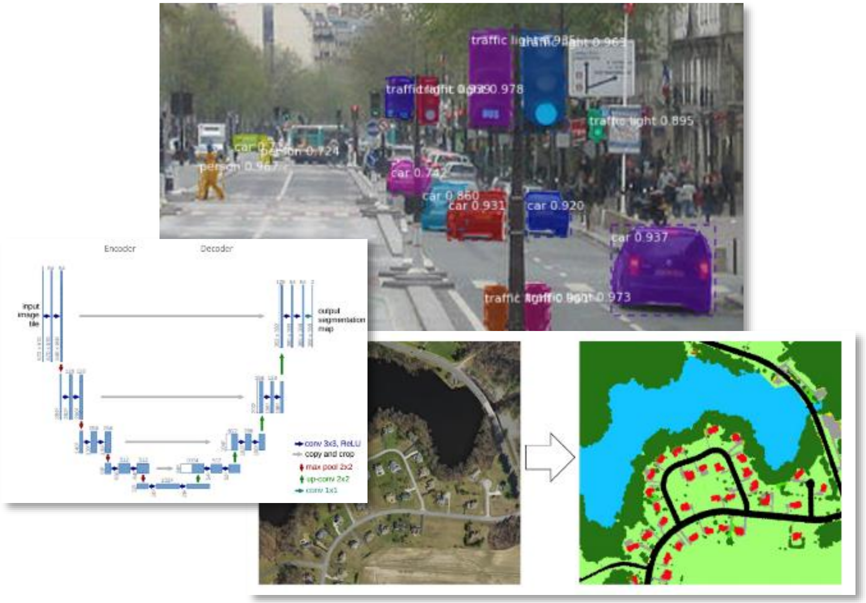
- works with the efficient KEA-file format (HDF5-based)
- keeps inode usage on NeSI HPC at a minimum
- includes cross-validation statistics and error visualization
- uses in-memory, tile-based prediction routines

- **Instance segmentation** of objects

  using MaskRCNN

- **Semantic segmentation** of

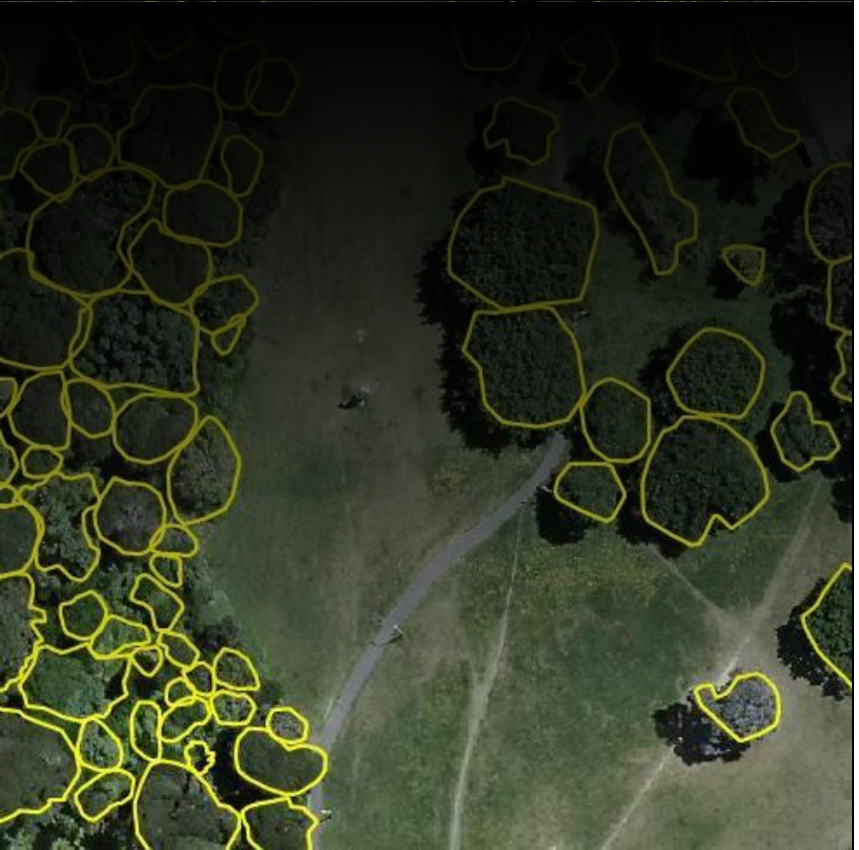
  using UNet64 and ResUNet



https://github.com/matterport/Mask_RCNN
https://arxiv.org/abs/1505.04597

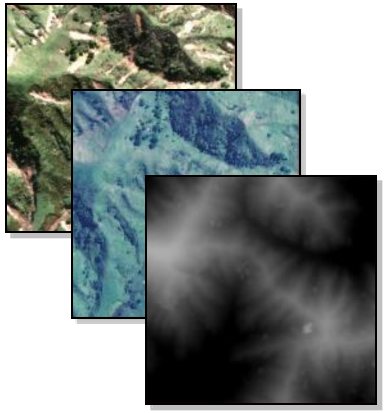
DeepSeg

Semantic and instance segmentation for remote sensing imagery

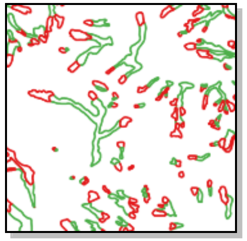


Overall workflow

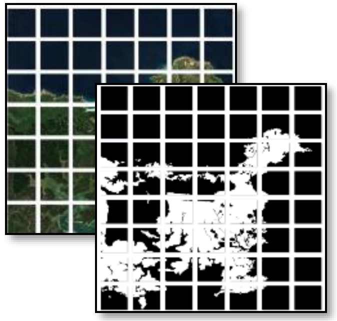
Image stack



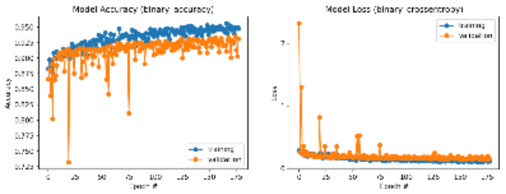
Label file



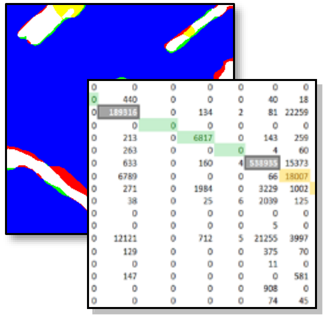
Data Preparation



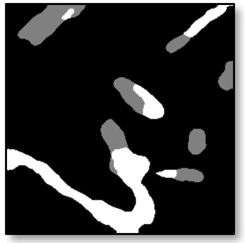
Training



Cross-Validation

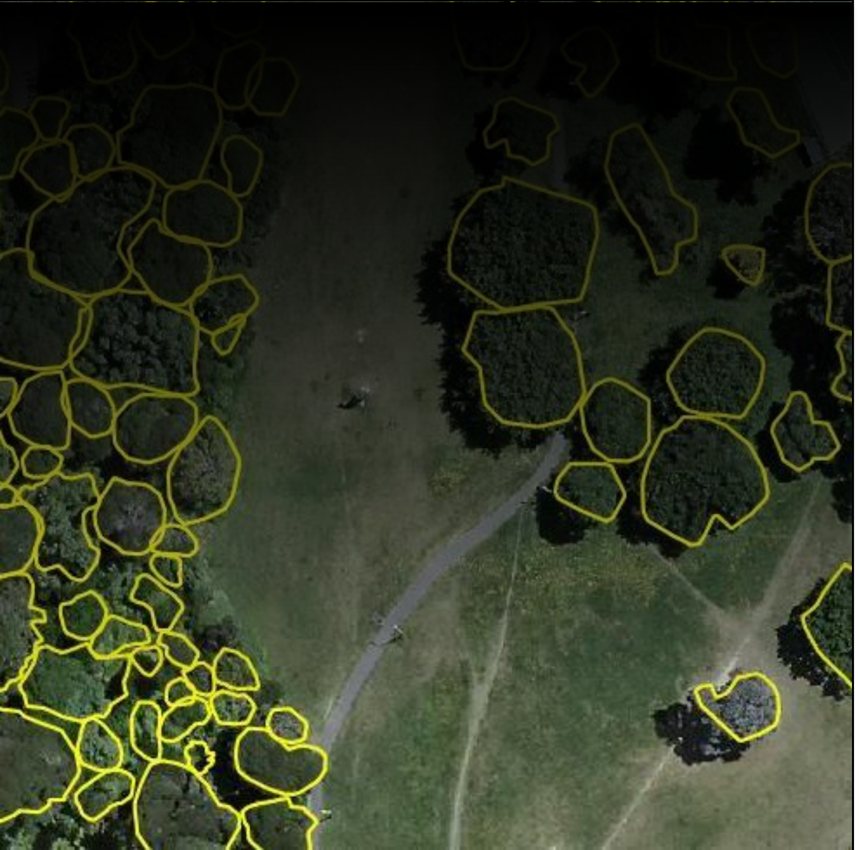


Prediction



DeepSeg

Semantic and instance segmentation for remote sensing imagery

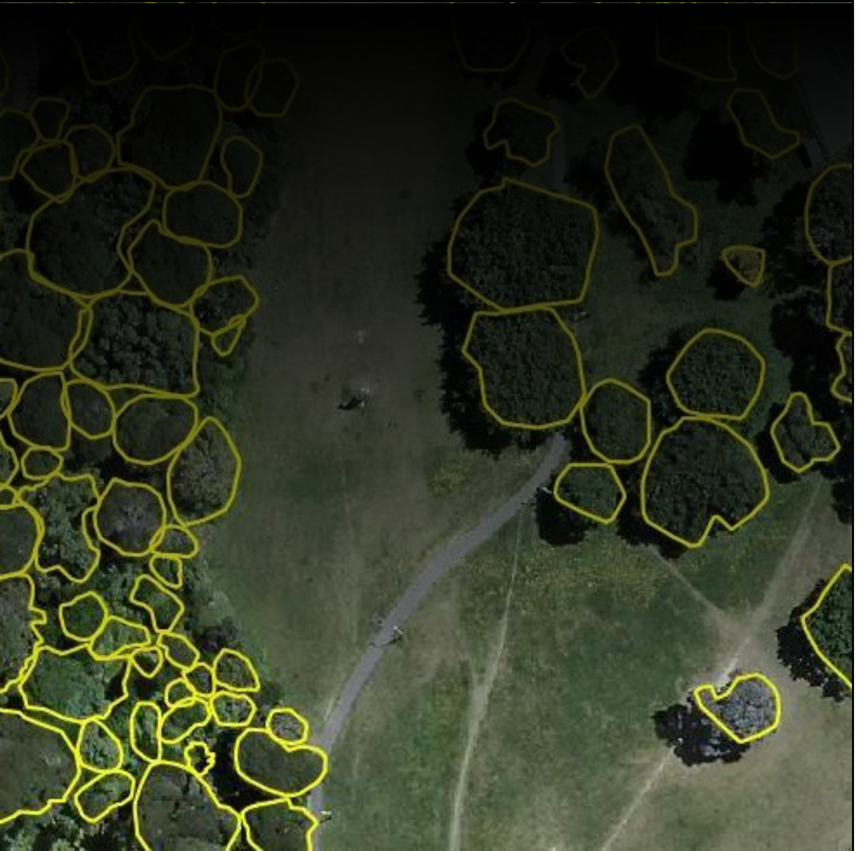


Easy installation on local PC, NeSI HPC, Singularity or Docker container

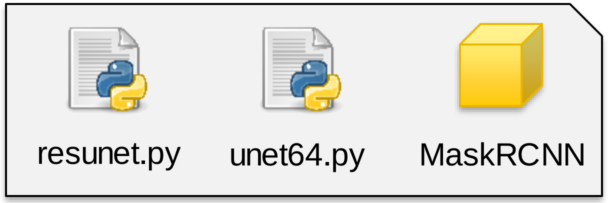
```
# Unload all modules (only on NeSI)
module purge

# Install Miniconda
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
sh Miniconda3-latest-Linux-x86_64.sh  # follow install instructions

# Create Environment
conda deactivate
conda create --name deepseg -c conda-forge python=3.8 -y
conda activate deepseg

pip install tensorflow==2.4.1 keras==2.4.3 aiohttp attrs chardet

# only needed for local PC - on NeSI we can load the appropriate modules
conda install cudnn=7.6.5 cudatoolkit=11.0 -c conda-forge

conda install pyyaml ipython geopandas rasterio tqdm ipdb cudnn rios \
        gdal Pillow cython scikit-image imgaug -c conda-forge –y

pip install opencv-python h5py==2.10.0 opencv-python-headless pycm runstats

# Install MaskRCNN
cd DeepSeg/models/Mask_RCNN-tf2
python setup.py install
```

DeepSeg

Semantic and instance segmentation for remote sensing imagery

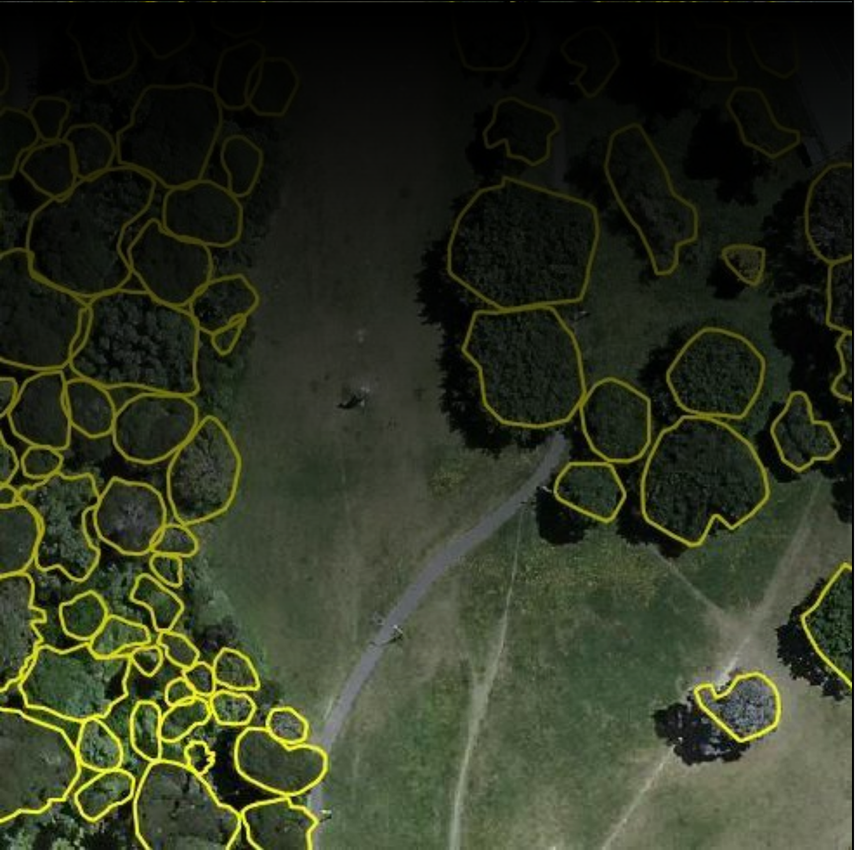


- **12 files**, 1 module, no more than 200 lines per file
- User **only needs to edit 1 file**: experiment.yaml
- User can **easily create more experiments** with different configurations
- Extensible **model zoo:**

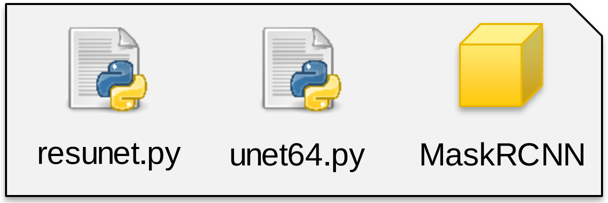


DeepSeg

Semantic and instance segmentation for remote sensing imagery



- **12 files**, 1 module, no more than 200 lines per file
- User **only needs to edit 1 file**: experiment.yaml
- User can **easily create more experiments** with different configurations
- Extensible **model zoo:**



experiment.yaml

```
# Configuration for DeepSeg

### General Parameters
resolution:        0.1
tilesize:        1024
overlap:          256
nbands:             3
…
segmentation_type: "instance"
stages:        “dataprep,train,crossval,predict“
…
epochs:          500
learning_rate:   0.001
…
```

DeepSeg

Semantic and instance segmentation for remote sensing imagery

- **12 files**, 1 module, no more than 200 lines per file
- User **only needs to edit 1 file**: experiment.yaml
- User can **easily create more experiments** with different configurations
- Extensible **model zoo:**

resunet.py unet64.py MaskRCNN

experiment.yaml

Orchestration

DeepSeg.sh

Data preparation

create_training_data.py

Training

train_instance.py train_semantic.py

Cross-Validation

crossval.py

Prediction

predict.py

```
# Configuration for DeepSeg

### General Parameters
resolution:    0.1
tilesize:      1024
overlap:       256
nbands:        3
…
segmentation_type: "instance"
stages:        "dataprep,train,crossval,predict"
…
epochs:        500
learning_rate: 0.001
…
```

DeepSeg

Semantic and instance segmentation for remote sensing imagery

Orchestration

DeepSeg.sh

Data preparation

create_training_data.py

Training

train_instance.py train_semantic.py

Cross-Validation

crossval.py

Prediction

predict.py

Execute DL pipeline locally / interactively or as batch job on NeSI

Run whole experiment (tile creation, training, cross-validation, prediction):

```
# locally
conda activate deepseg
sh DeepSeg.sh YOUR-EXPERIMENT

# on NeSI
sbatch DeepSeg.sh YOUR-EXPERIMENT
```

```
...
SBATCH --job-name=DeepSeg
SBATCH --partition=gpu
SBATCH --gpus-per-node=A100:1
module load CUDA/11.2.0
...
```

Run individual steps with:

```
export DEEPSEG_EXPERIMENT=YOUR-EXPERIMENT
conda activate deepseg
python create_training_data.py
python train_semantic.py
python predict.py
```

DeepSeg

Semantic and instance segmentation for remote sensing imagery

Orchestration

DeepSeg.sh

Data preparation

create_training_data.py

Training

train_instance.py train_semantic.py

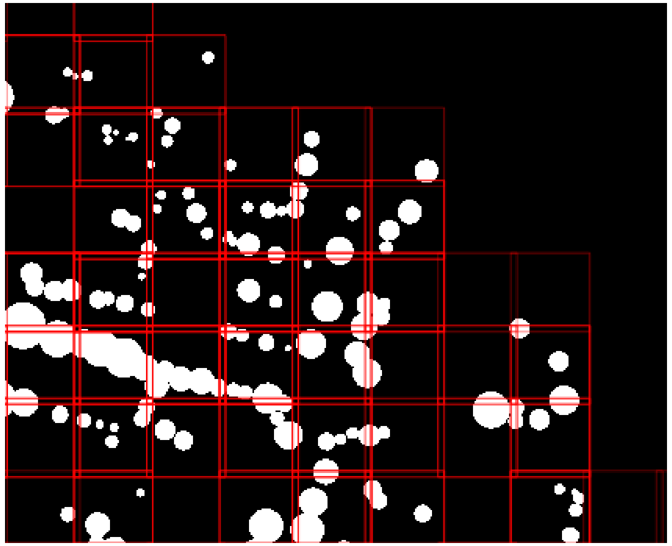
Cross-Validation

crossval.py

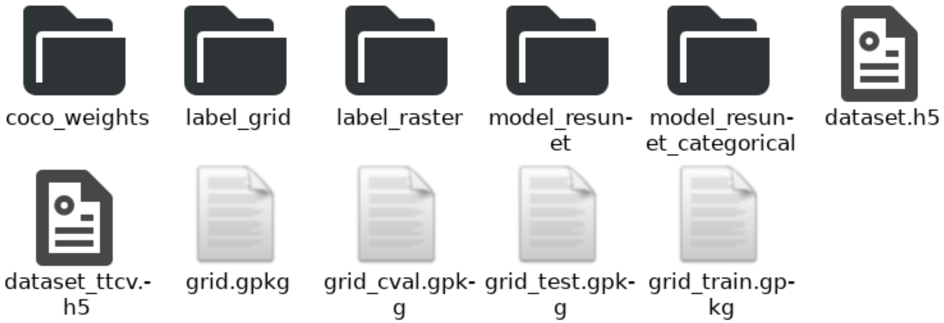
Prediction

predict.py

- Rasterization of vector labels
- Creation of (overlapping) vector tiles
- Intersection with vector or raster labels



- Storage of image and label tiles in **one** HDF5 file
- HDF5 data file can be re-used in multiple experiments
- Calculation of online statistics for datasets larger than memory



DeepSeg

Semantic and instance segmentation for remote sensing imagery

Orchestration
DeepSeg.sh

Data preparation
create_training_data.py

Training
train_instance.py train_semantic.py

Cross-Validation
crossval.py

Prediction
predict.py

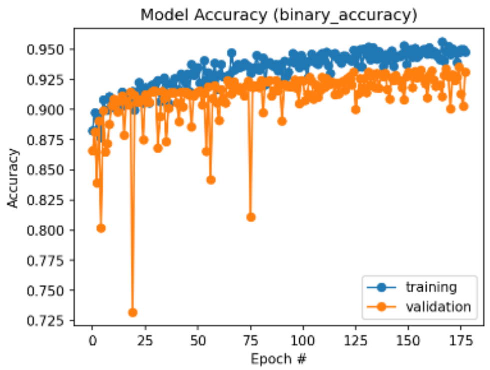
- **Flexible model training for instance or semantic models** with Tensorflow-based routines
- **Custom image generator class** to ingest image/label tiles from HDF5 file

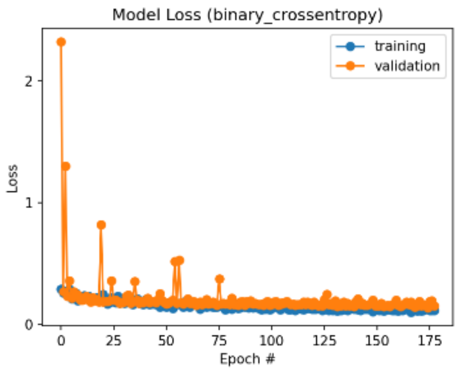
```
def HDF5ImageGenerator(hdf5_x, hdf5_y, batch_size, statistics):
  …
```

- MaskRCNN required **refactoring to work in TF v2.4**
- **Custom Dataset class for MaskRCNN** to work with the same HDF5 data file

```
class HDF5Dataset(utils.Dataset):
   def load_hdf5(self, dataset_source, subset):
      """ loads the image tiles from the prepared HDF5 file
      """
      …
```

- Careful **image augmentation** (e.g., avoid smoothing effects during rotation)
- **Immediate training feedback** with log graphs (image files) in addition to Tensorboard





Classification error visualization

True Positive
False Positive
True Negative
False Negative

Orchestration

DeepSeg.sh

Data preparation

create_training_data.py

Training

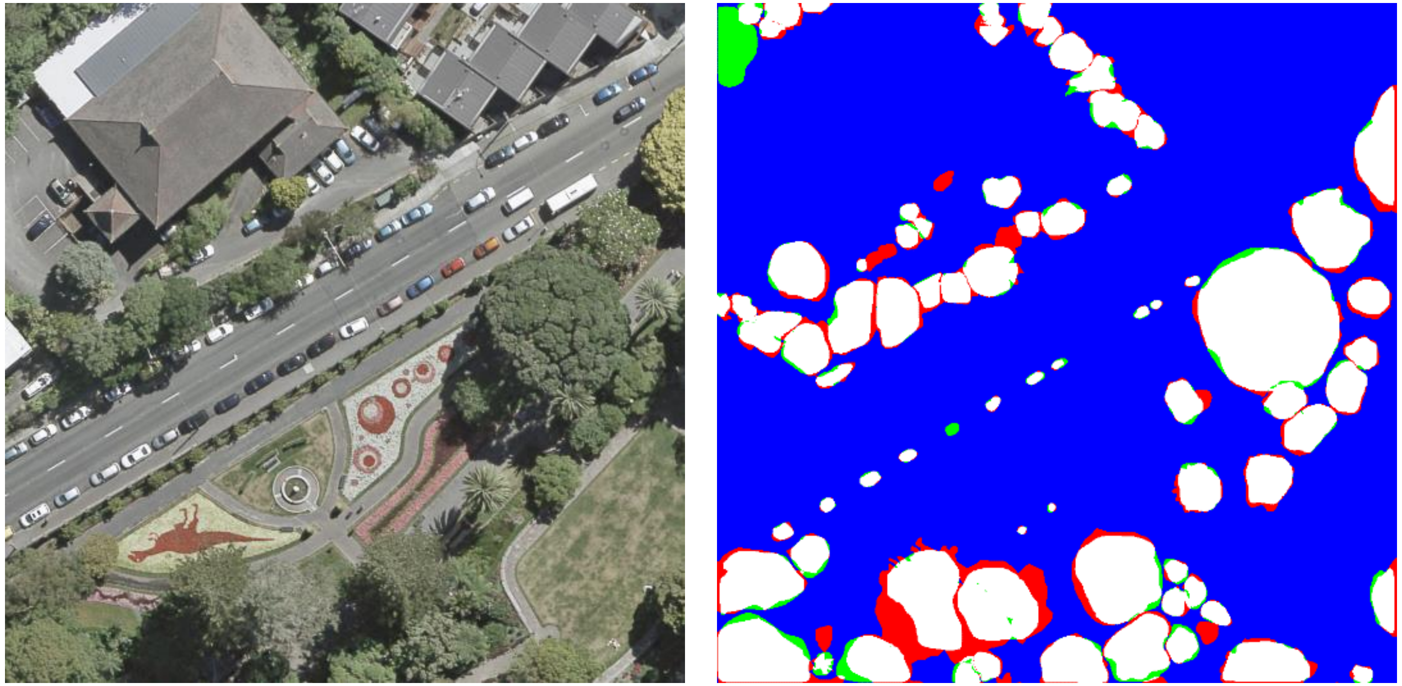
train_instance.py train_semantic.py

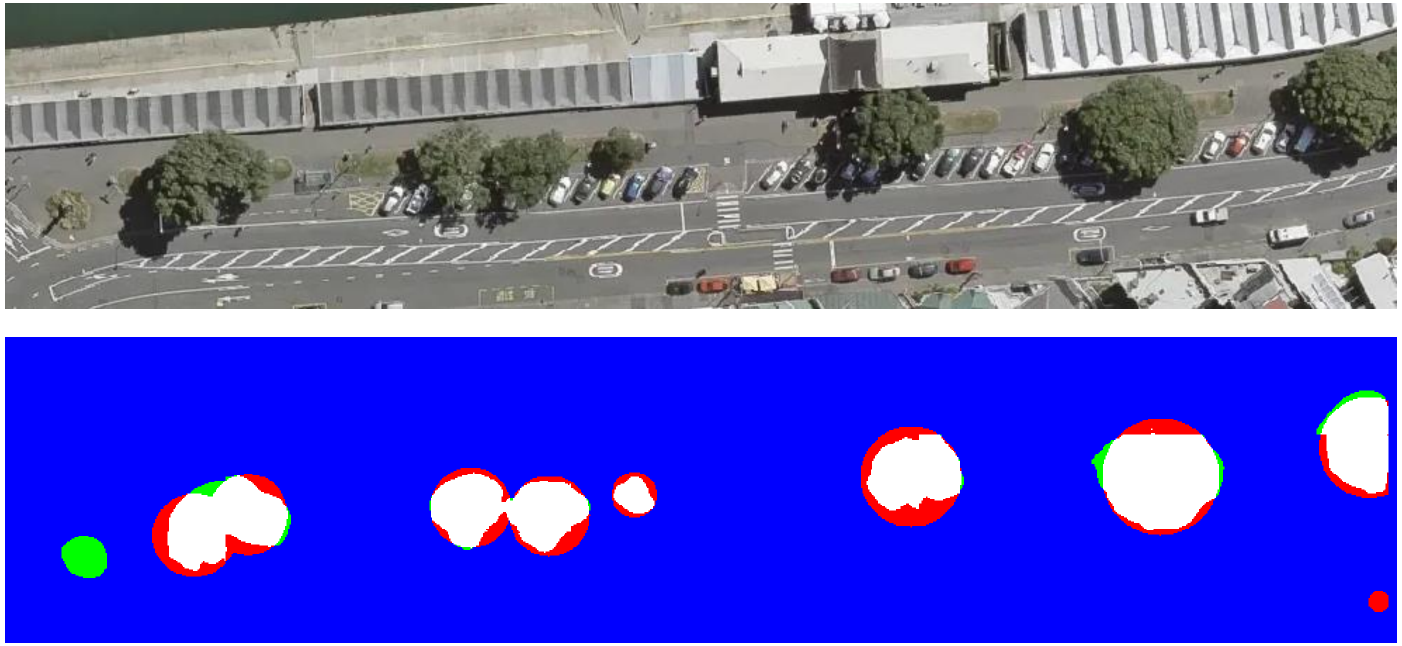
Cross-Validation

crossval.py

Prediction

predict.py





Detailed accuracy statistics / reports

Orchestration
DeepSeg.sh

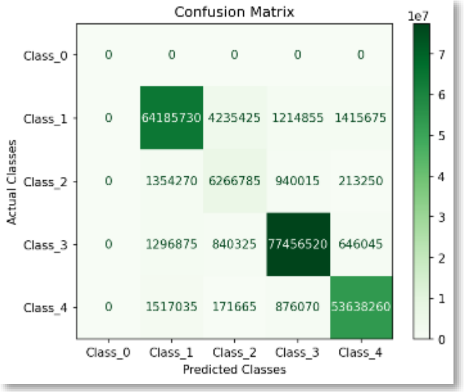
Data preparation
create_training_data.py

Training
train_instance.py

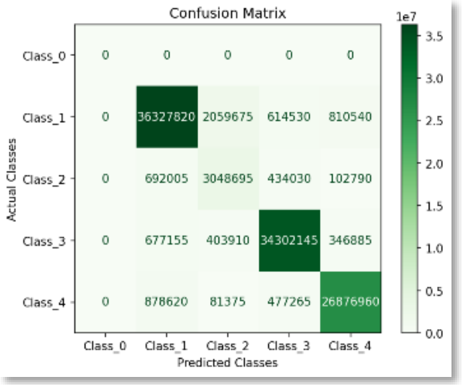
Cross-Validation
crossval.py

Prediction
predict.py

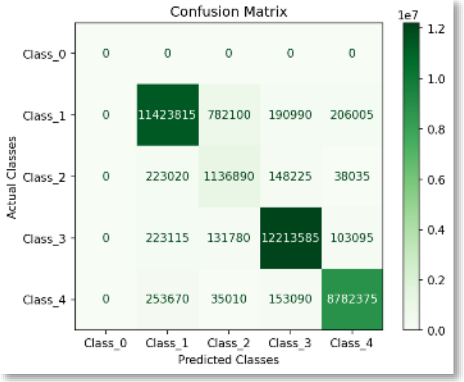
Training



Test



Cross-Validation



Class Statistics :

Class	Class_0	Class_1	Class_2	Class_3	Class_4	Description
ACC	1.0	0.94898	0.96414	0.97312	0.97762	Accuracy
AGF	None	0.93338	0.81378	0.9719	0.96975	Adjusted F-score
AGM	None	0.95061	0.90308	0.9739	0.97669	Adjusted geometric
AM	0	-2697775	2739880	247695	-289800	Difference between
AUC	None	0.93733	0.84446	0.97152	0.97008	Area under the RO
AUCI	None	Excellent	Very Good	Excellent	Excellent	AUC value interpre
AUPR	None	0.92119	0.62924	0.96383	0.95684	Area under the PR
BCD	0.0	0.00624	0.00633	0.00057	0.00067	Bray-Curtis dissimi
BM	None	0.87466	0.68893	0.94303	0.94015	Informedness or bo
CEN	None	0.15957	0.41847	0.08845	0.09757	Confusion entropy
DOR	None	316.34493	96.32402	1221.16578	1450.54852	Diagnostic odds rat
DP	None	1.37841	1.09369	1.70183	1.74304	Discriminant power
DPI	None	Limited	Limited	Limited	Limited	Discriminant power
ERR	0.0	0.05102	0.03586	0.02688	0.02238	Error rate
F0.5	None	0.93167	0.57146	0.96294	0.95832	F0.5 score
F1	None	0.92085	0.61777	0.96383	0.95683	F1 score - harmonic
F2	None	0.91028	0.67224	0.96472	0.95535	F2 score
FDR	None	0.06098	0.45573	0.03766	0.04069	False discovery rate
FN	0	6865955	2507535	2783245	2564770	False negative/miss
FNR	None	0.09663	0.28578	0.03469	0.04563	Miss rate or false ne
FOR	0.0	0.04642	0.01225	0.0205	0.01599	False omission rate
FP	0	4168180	5247415	3030940	2274970	False positive/type
FPR	0.0	0.0287	0.02529	0.02228	0.01421	Fall-out or false pos
G	None	0.92102	0.62348	0.96383	0.95684	G-measure geometr

Class Statistics :

Class	Class_0	Class_1	Class_2	Class_3	Class_4	Description
ACC	1.0	0.94699	0.9651	0.97268	0.97505	Accuracy
AGF	None	0.93557	0.81343	0.96969	0.96632	Adjusted F-score
AGM	None	0.95012	0.90323	0.97324	0.97406	Adjusted geometric mean
AM	0	-1236965	1316135	97875	-177045	Difference between automatic ar
AUC	None	0.93979	0.84411	0.96948	0.96673	Area under the ROC curve
AUCI	None	Excellent	Very Good	Excellent	Excellent	AUC value interpretation
AUPR	None	0.9271	0.62888	0.95872	0.95223	Area under the PR curve
BCD	0.0	0.00572	0.00609	0.00045	0.00082	Bray-Curtis dissimilarity
BM	None	0.87957	0.68822	0.93896	0.93345	Informedness or bookmaker infc
CEN	None	0.15102	0.41768	0.09829	0.10497	Confusion entropy
DOR	None	306.44003	98.7651	1115.88116	1165.73931	Diagnostic odds ratio
DP	None	1.37079	1.09968	1.68024	1.69071	Discriminant power
DPI	None	Limited	Limited	Limited	Limited	Discriminant power interpretatic
ERR	0.0	0.05301	0.0349	0.02732	0.02495	Error rate
F0.5	None	0.93573	0.57194	0.95794	0.95401	F0.5 score
F1	None	0.92687	0.6177	0.95872	0.95222	F1 score - harmonic mean of pre
F2	None	0.91818	0.67141	0.95951	0.95043	F2 score
FDR	None	0.05827	0.45497	0.04259	0.04479	False discovery rate
FN	0	3484745	1228825	1427950	1437260	False negative/miss/type 2 error
FNR	None	0.08753	0.28728	0.03996	0.05076	Miss rate or false negative rate
FOR	0.0	0.0501	0.01198	0.01975	0.01797	False omission rate
FP	0	2247780	2544960	1525825	1260215	False positive/type 1 error/false
FPR	0.0	0.0329	0.0245	0.02107	0.01579	Fall-out or false positive rate
G	None	0.92699	0.62326	0.95872	0.95222	G-measure geometric mean of p

Class Statistics :

Class	Class_0	Class_1	Class_2	Class_3	Class_4	Description
ACC	1.0	0.94787	0.96232	0.97364	0.97811	Accuracy
AGF	None	0.93374	0.82273	0.97168	0.96905	Adjusted F-score
AGM	None	0.95052	0.90767	0.97435	0.97695	Adjusted geometric mean
AM	0	-479290	539610	34315	-94635	Difference between automatic and manual classification
AUC	None	0.93829	0.85389	0.9714	0.96958	Area under the ROC curve
AUCI	None	Excellent	Very Good	Excellent	Excellent	AUC value interpretation
AUPR	None	0.92436	0.64018	0.96256	0.95704	Area under the PR curve
BCD	0.0	0.00665	0.00749	0.00048	0.00131	Bray-Curtis dissimilarity
BM	None	0.87659	0.70779	0.94279	0.93916	Informedness or bookmaker informedness
CEN	None	0.15323	0.40653	0.09088	0.09779	Confusion entropy
DOR	None	314.85863	98.21351	1239.44255	1516.10417	Diagnostic odds ratio
DP	None	1.37728	1.09834	1.70539	1.75363	Discriminant power
DPI	None	Limited	Limited	Limited	Limited	Discriminant power interpretation
ERR	0.0	0.05213	0.03768	0.02636	0.02189	Error rate
F0.5	None	0.93489	0.57481	0.96177	0.95999	F0.5 score
F1	None	0.92401	0.62605	0.96255	0.95702	F1 score - harmonic mean of precision and sensitivity
F2	None	0.91339	0.68732	0.96334	0.95406	F2 score
FDR	None	0.05772	0.45493	0.03875	0.03802	False discovery rate
FN	0	1179095	409280	457990	441770	False negative/miss/type 2 error
FNR	None	0.09356	0.26471	0.03614	0.04789	Miss rate or false negative rate
FOR	0.0	0.04929	0.01205	0.01962	0.01641	False omission rate
FP	0	699805	948890	492305	347135	False positive/type 1 error/false alarm
FPR	0.0	0.02985	0.02751	0.02106	0.01294	Fall-out or false positive rate
G	None	0.92419	0.63308	0.96255	0.95703	G-measure geometric mean of precision and sensitivity

Prediction pipeline

Orchestration

DeepSeg.sh

Data preparation

create_training_data.py

Training

train_instance.py

Cross-Validation

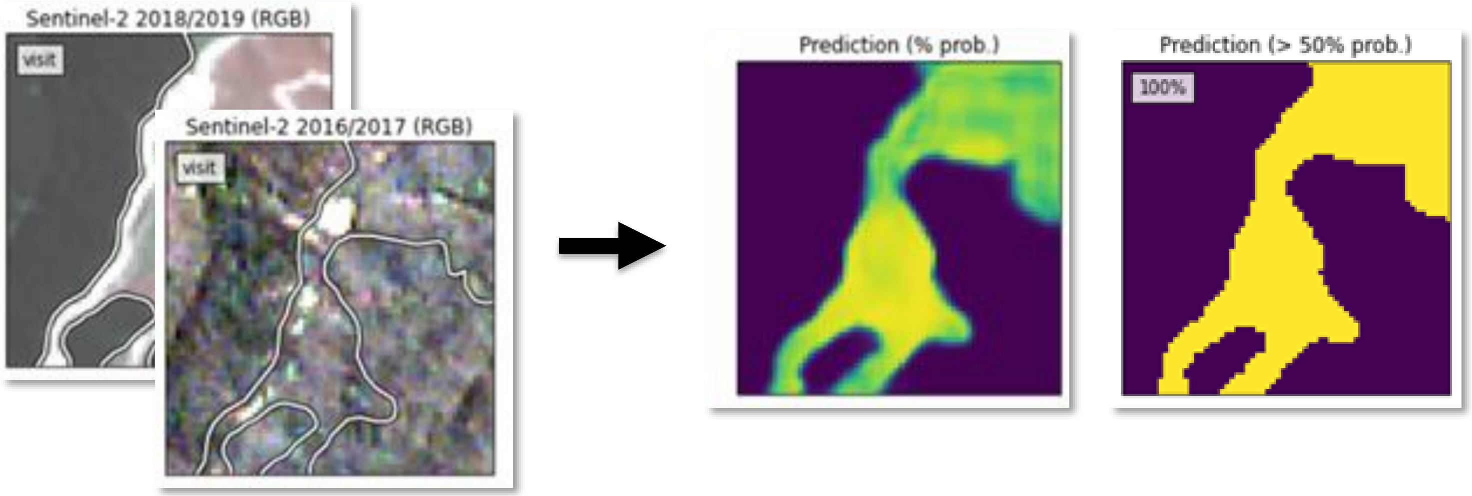
crossval.py

Prediction

predict.py

Experiences and recommendations

- The same prediction routine for instance and semantic classification
- GDAL/rasterio-based, on-the-fly processing of extremely large raster files
- Batch size depends on memory requirements (how many tiles are processed at the same time)
- Tile overlap should be chosen based on the largest objects
- Consider simplifying polygons to avoid extremely large vector-files
- Pre-defined colormaps (or random colours) are stored with the prediction



Experiments

Individual tree detection and instance segmentation

Training

446 tiles at 1024 x 1024 px

Prediction

Raster 146,400 x 237,600 px

59,210 tiles at 1024 x 1024 px

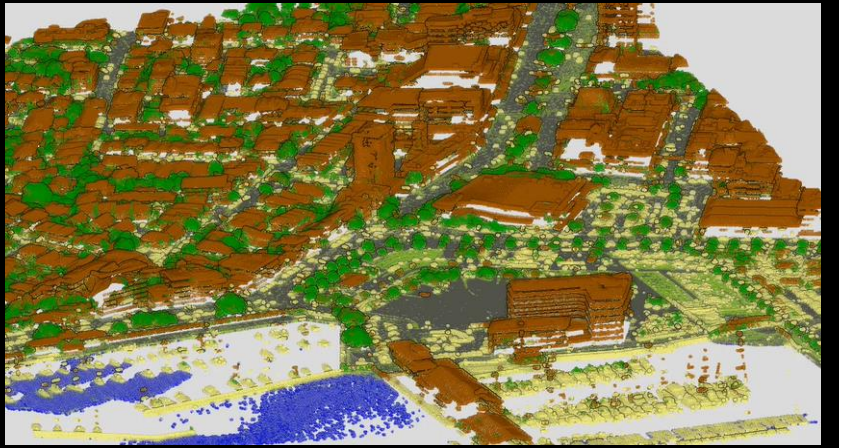
(256 px overlap)

Number of instances:

~670,000 trees

Experiments

Multi-class
semantic classification

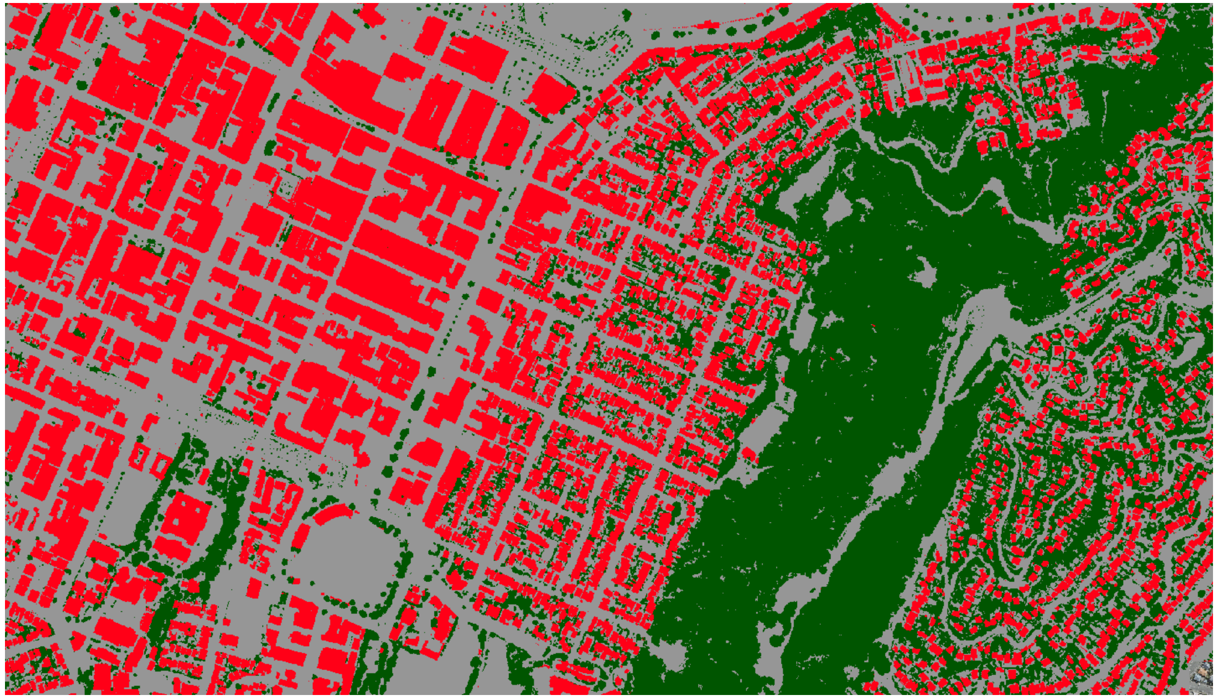


Training on LiDAR-derived classes

Ground

Tree Canopy

Buildings

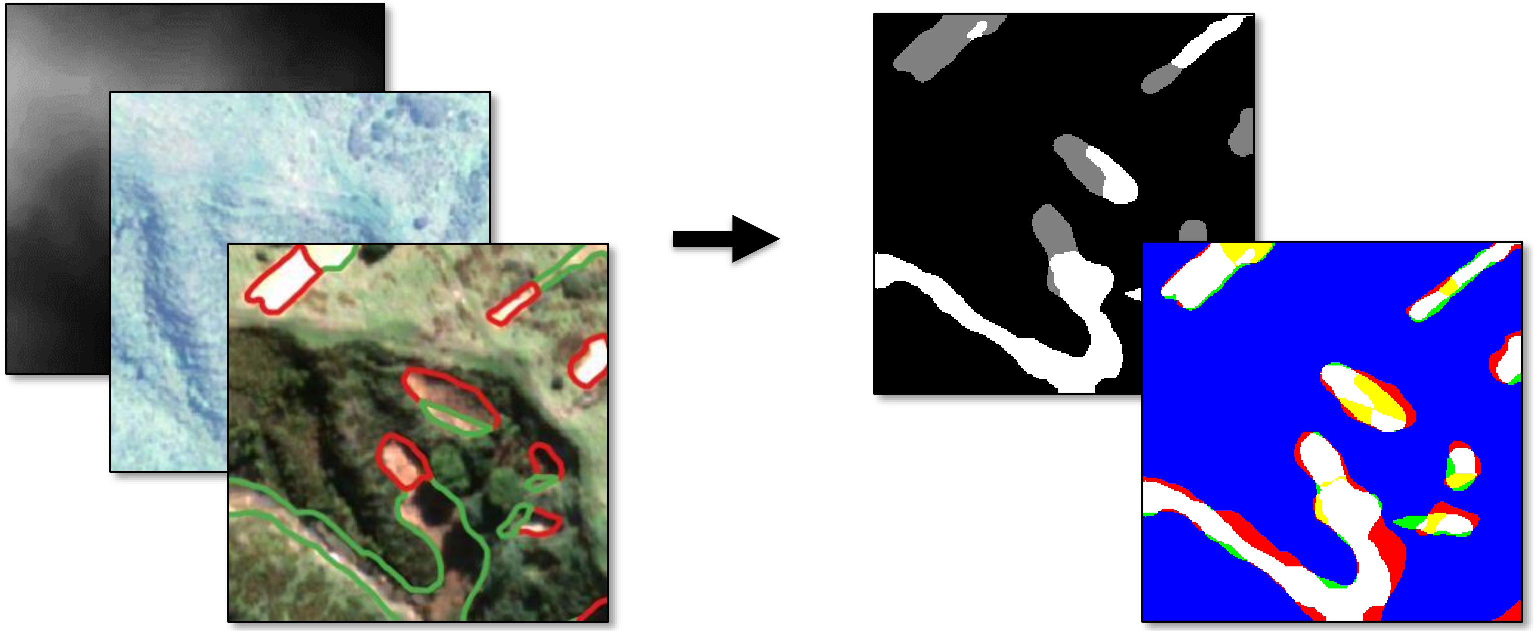


Experiments

Mapping land slides

Inputs:

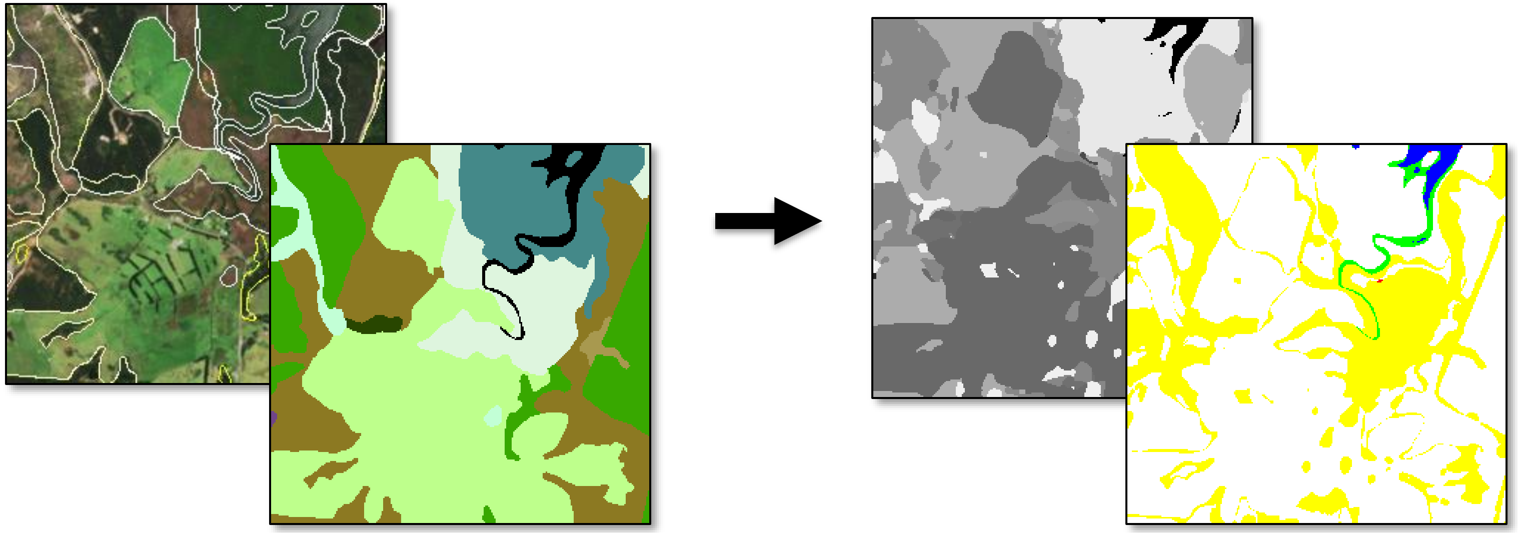
- Digital Elevation Model
- 4-Band WorldView (before)
- 4-Band WorldView (after)



Mapping land cover

Inputs:

- 10-band Sentinel-2 imagery
- LCDB 5 spatial layer

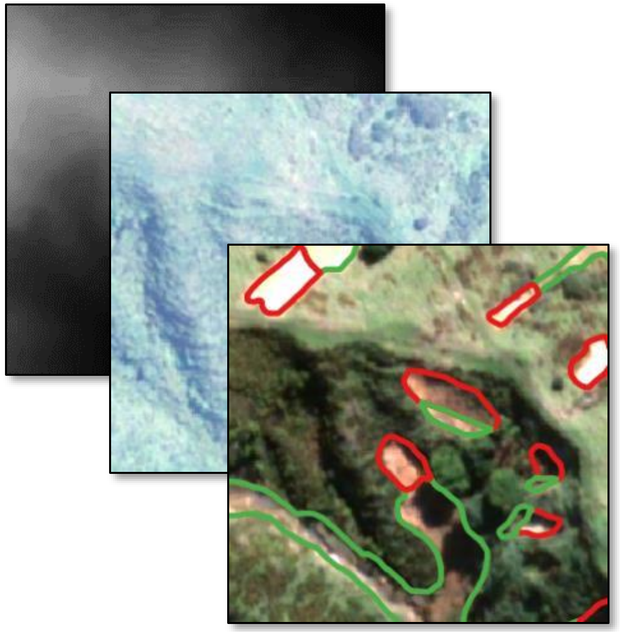


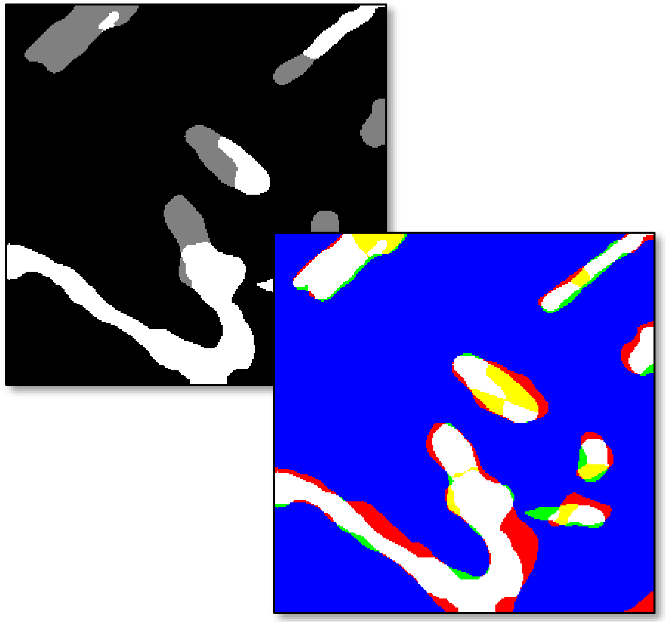
Experiments

Mapping land slides

Inputs:

- Digital Elevation Model
- 4-Band WorldView (before)
- 4-Band WorldView (after)

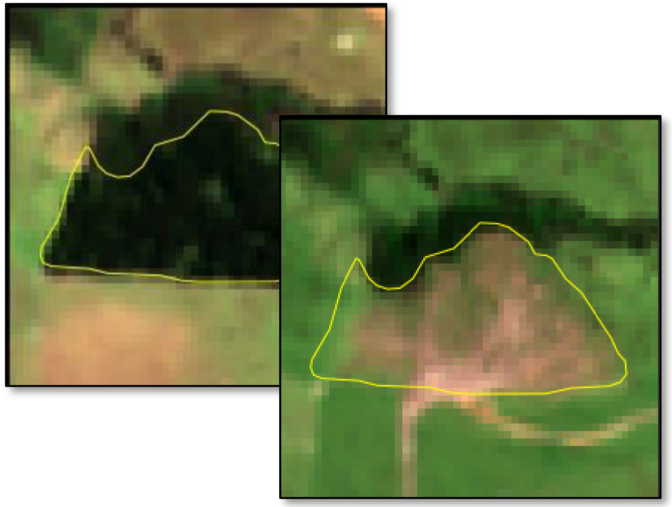


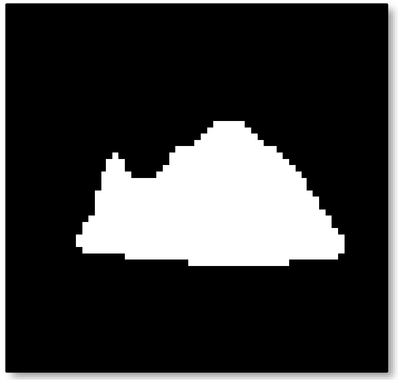


Mapping forest destock

Inputs:

- 10-Band Sentinel-2 (before)
- 10-Band Sentinel-2 (after)





Accuracy: 95%

Most errors are too difficult for humans to distinguish

Thank you

Questions?

Feel free to contact me at schindlerj@landcareresearch.co.nz

or our team at www.landcareresearch.co.nz