Deep Learning for Glacier Mapping

Finse Alpine Research Centre, Norway

Konstantin A. Maslov

University of Twente,
Faculty of Geo-Information Science
and Earth Observation

Thomas Schellenberger

University of Oslo,
Faculty of Mathematics and Natural
Sciences

September 2022

Outline

Image classification

Image classification (in the context of remote sensing) or semantic image segmentation (in the context of machine learning/deep learning) is a problem of segmenting the whole digital image into semantically meaningful classes.

There several types of features that help to classify objects in remotely sensed images:

- spectral
- textural
- shapes
- topological

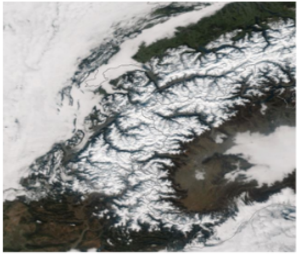




Image classification

In general, one can identify three approaches to address the classification problem:

- Pixel-to-pixel (the majority of the classification methods; can take into account only the spectral information)

$$\mathbf{y_{ij}} = f(\mathbf{x_{ij}}) + \epsilon \tag{1}$$

- Patch-to-pixel (textural analysis, convolutional neural networks, transformers)

$$\mathbf{y_{ij}} = f(\mathbf{X_{i-i+h,j-j+w}}) + \epsilon \tag{2}$$

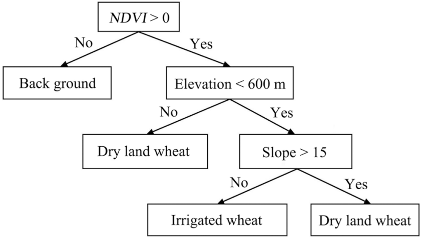
- Patch-to-patch (fully-convolutional neural networks, transformers)

$$\mathbf{Y_{i-i+h,j-j+w}} = f(\mathbf{X_{i-i+h,j-j+w}}) + \epsilon \tag{3}$$

Random forest

Random forest—an ensemble of decision trees trained to solve one problem.

Decision trees:



Source: Meng et al., 2014

- No-leaf nodes are splits (rules to build decision boundaries)
- Leaf nodes are decisions

- To make a split, compare how every feature threshold reduces 'impurity', e.g., Gini impurity
- $gini_{leaf} = 1 - p_0^2 - p_1^2$
- $gini_{split} = \frac{N_{yes}}{N_{yes}+N_{no}} gini_{yesleaf} + \frac{N_{no}}{N_{yes}+N_{no}} gini_{noleaf}$
- At each split, choose a threshold to minimize $gini_{split}$ in a greedy manner
- After every leaf is 'pure', assign labels to the leaves

Random forest

The output of a random forest model is the average prediction of the decision trees:

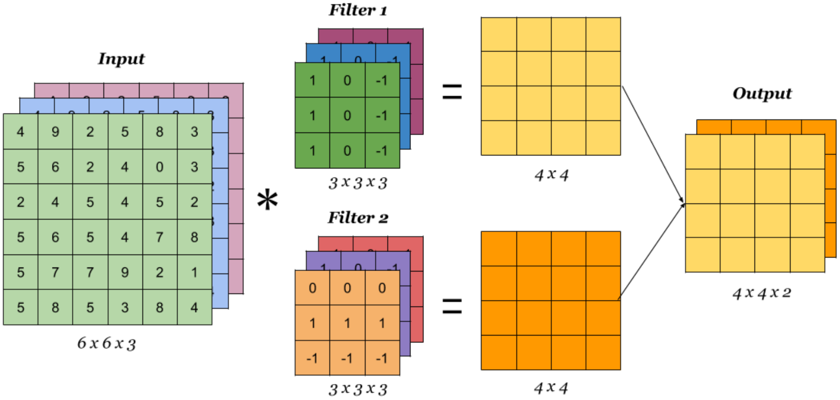
$$forest(\mathbf{x}) = \frac{1}{N_{trees}} \sum_i tree_i(\mathbf{x}). \tag{4}$$

But we do not want the decision trees to be correlated, so

- use data bootstrapping
- use feature bootstrapping at each split

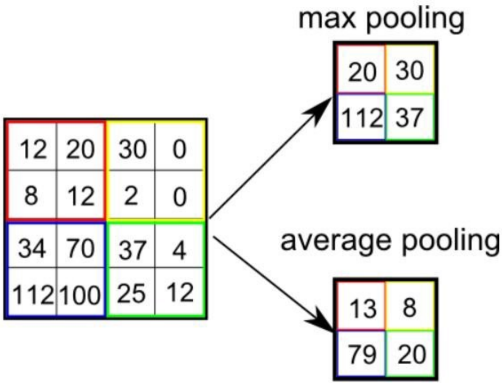
Fully-convolutional networks

Let's first recall what convolutional neural networks are.



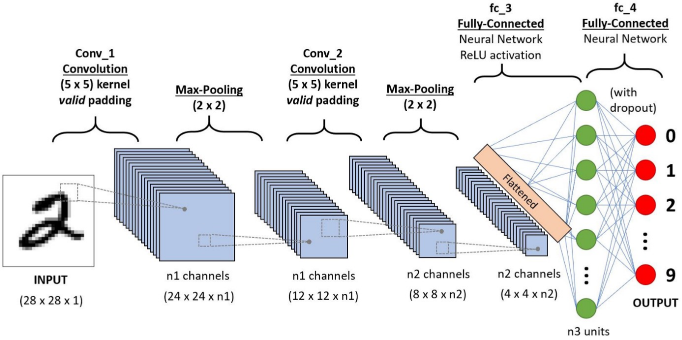
Source: `https://bit.ly/3eoHAJd`

Fully-convolutional networks



Source: `https://bit.ly/3L0BHhL`

Fully-convolutional networks

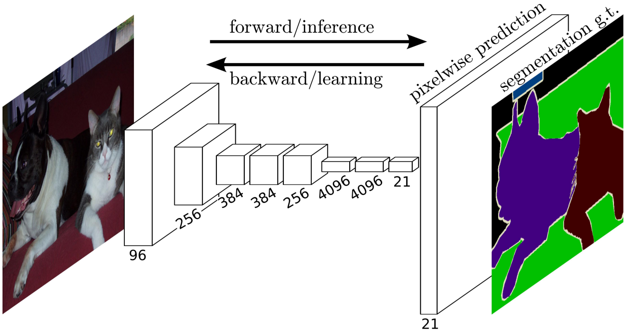


Source: `https://bit.ly/3wU3ryu`

- One inference to classify one pixel
- Computationally intensive
- Still can neglect some spatial correlations producing ‘noisy’ results

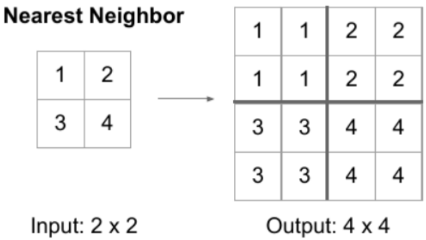
Fully-convolutional networks

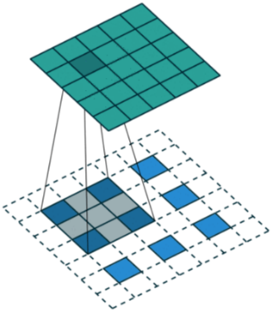
Can we do better? Yes!

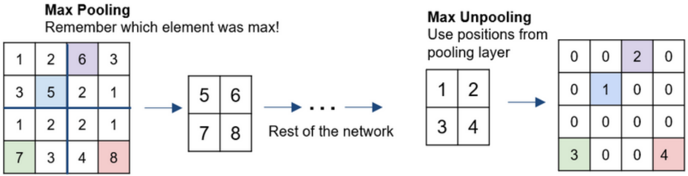


Source: Long et al., 2014

Fully-convolutional networks

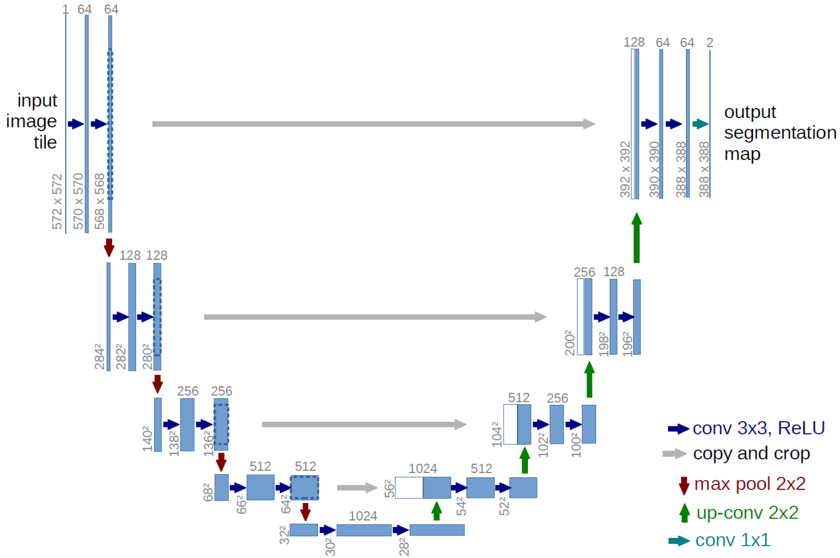






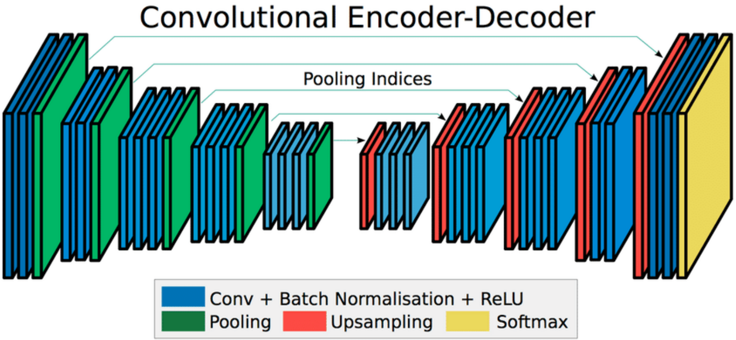
Source: `https://bit.ly/3RkuCL6`

Fully-convolutional networks



Source: Ronneberger et al., 2015

Fully-convolutional networks



Source: Badrinarayanan et al., 2015

Fully-convolutional networks

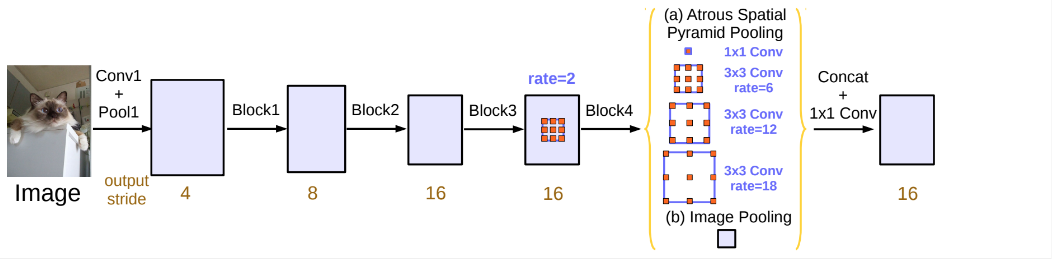
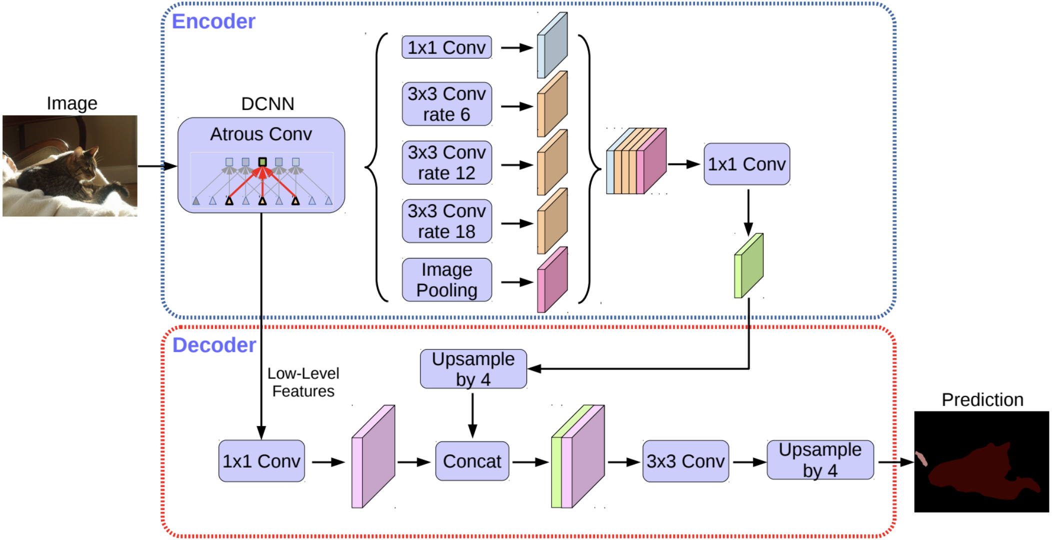


Figure 5. Parallel modules with atrous convolution (ASPP), augmented with image-level features.

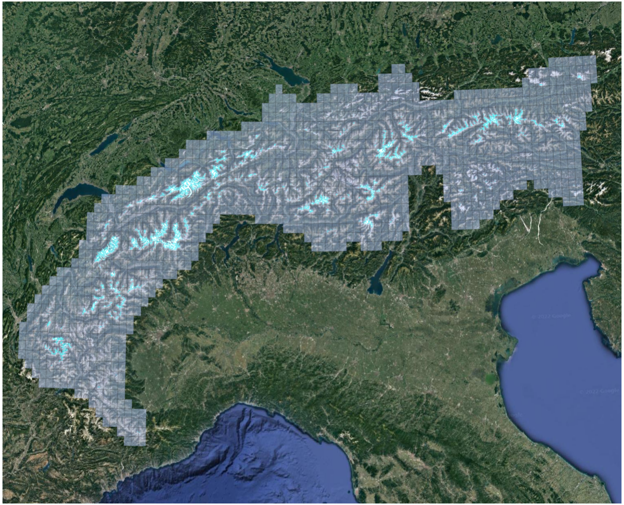
Source: Chen et al., 2017

Fully-convolutional networks



Source: Chen et al., 2018

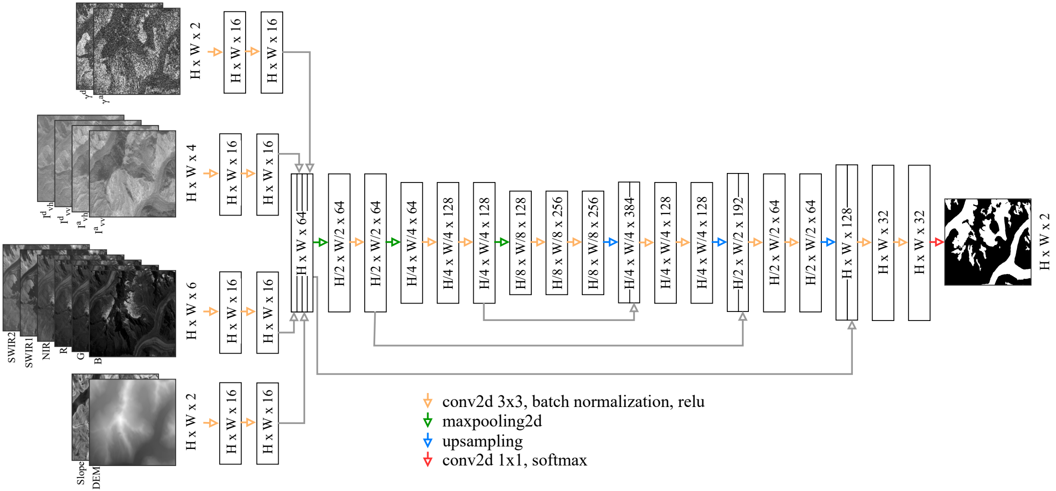
Case study: glacier mapping in the Alps



- While it is a relatively simple problem to map clean snow/ice, debris-covered parts present a great challenge
- Do U-Net–based methods have an advantage over simpler models such as random forests?
- How does including additional input features (DEM and SAR) affect the performance?

- As the 'groundtruth' data, we used the inventory by Paul et al., 2020
- As input features—Sentinel-1/2 imagery and Copernicus DEM

Case study: glacier mapping in the Alps

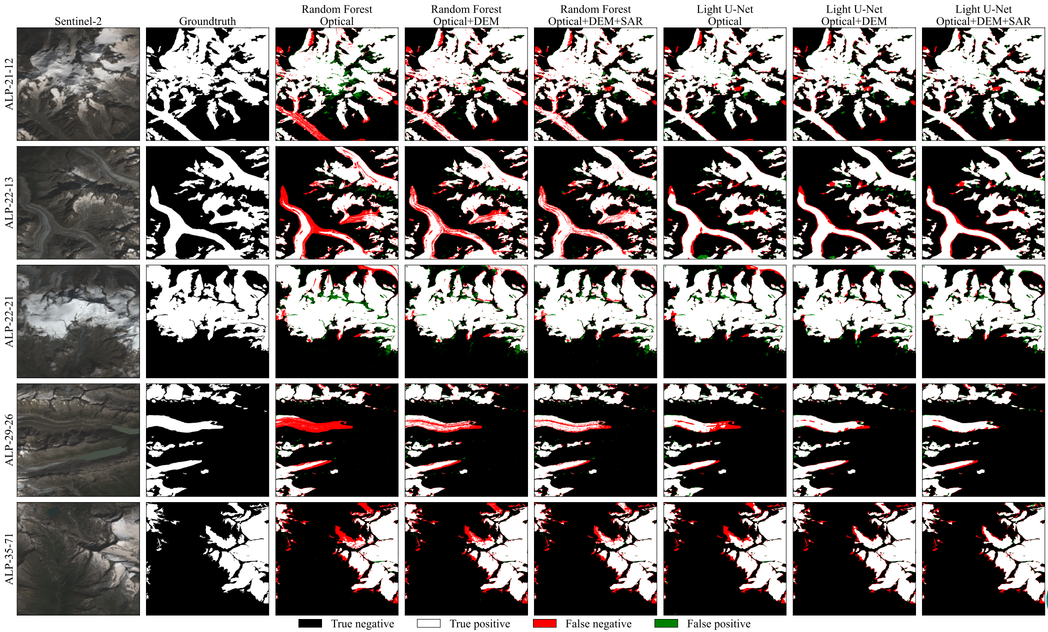


Case study: glacier mapping in the Alps

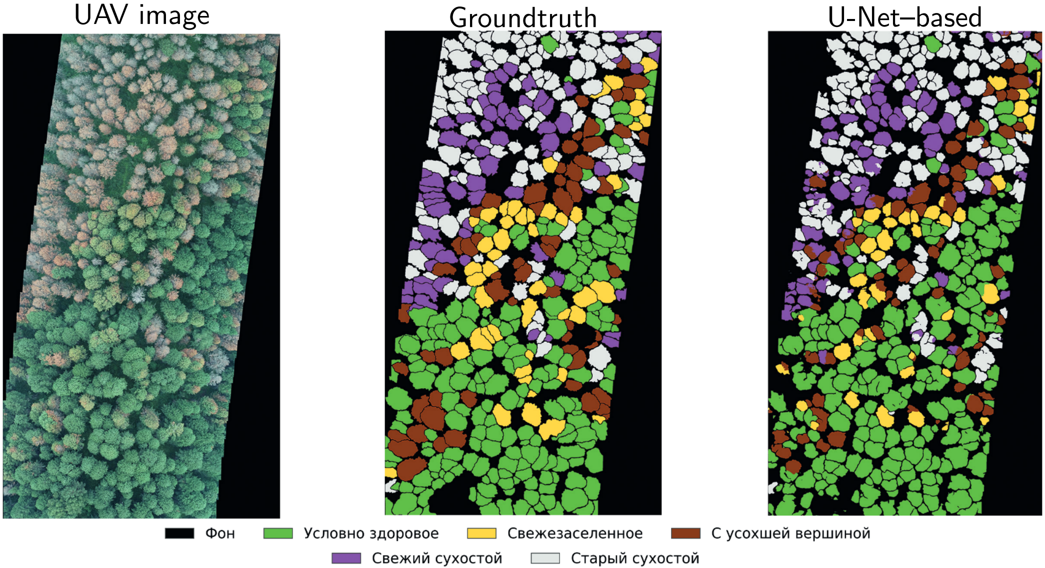
Data	Accuracy	Precision	Recall	F1-score	IoU
Random forest					
Optical	0.986	0.929	0.828	0.876	0.779
Optical+DEM	0.989	0.941	0.857	0.897	0.813
Optical+DEM+SAR	0.989	0.944	0.870	0.905	0.827
U-Net–based					
Optical	0.991	0.946	0.893	0.919	0.850
Optical+DEM	0.992	**0.950**	0.906	0.928	0.865
Optical+DEM+SAR	**0.992**	0.948	**0.917**	**0.932**	**0.873**

- U-Net–based methods outperform random forest
- Adding DEM and SAR data increases the performance (especially, for the glacier tongues)

Case study: glacier mapping in the Alps



Case study: mapping forests damaged by pests in Western Siberia



- Just to note that the methods can be relatively easily adapted for solving other problems

Project

In the project, we suggest going deeper with random forests and several deep learning models for glacier mapping in different world regions.

What we can offer for the beginning:

- A Jupyter notebook showing how to classify images with random forest
- A Jupyter notebook and a small python module to employ fully-convolutional networks for semantic image segmentation
- Three pretrained models—U-Net, SegNet, DeepLabV3+ (slightly modified)
- $\approx$ 200 GB dataset with optical imagery, SAR and DEM data that covers the Alps, Hunza valley (Northern Pakistan) and some areas of Svalbard

Project: dataset overview

	The Alps 2015	Northern Pakistan 2005	Northern Pakistan 2021	Svalbard 2020
Number of tiles	296	55	55	378
Optical (6 bands)	✓	✓	✓	✓
DEM and slope	✓	✓	✓	✓
Co-pol SAR intensity	✓	✓	✓	✓
Cross-pol SAR intensity	✓		✓	✓
Co-pol InSAR coherence	✓		✓	✓
Groundtruth	✓	✓		✓

- All the data are already split into train, val and test subsets
- For a better overview of the dataset, please see the `/data/overview` directory on Hub

Project: possibilities

- Tweaking the deep learning models
- Modifying the training routines
- Hyperparameter tuning for random forest
- Labelling snow cover data and training multi-class classification models
- Exploring how different feawture sets affect classification performance
- Change detection in Northern Pakistan

- Do not hesitate to suggest your ideas!