Semantic Segmentation of RGB-Z Aerial Imagery Using Convolutional Neural Networks

Amber E. Mulder 2020

Supervisors: Balázs Dukai & Ravi Peters
Co-reader: Jantien Stoter
Company supervisors: Sven Briels & Jean-Michel Renders

TU Delft

READAR
real estate radar

Content

Motivation

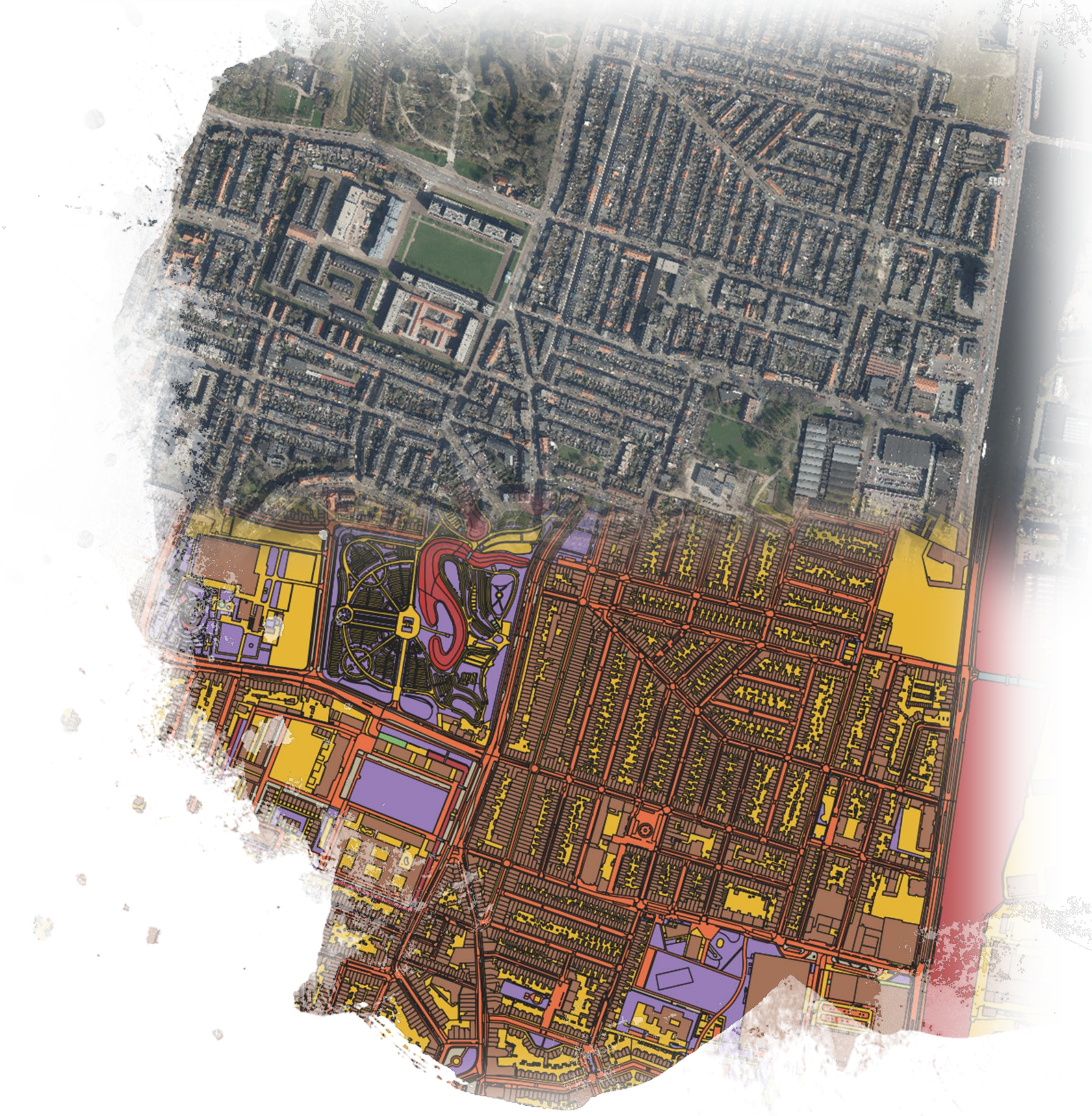
Motivation

Semantic segmentation

- Mapping of land cover
- Object detection
- Change detection
- Etc.

Example: BGT updating

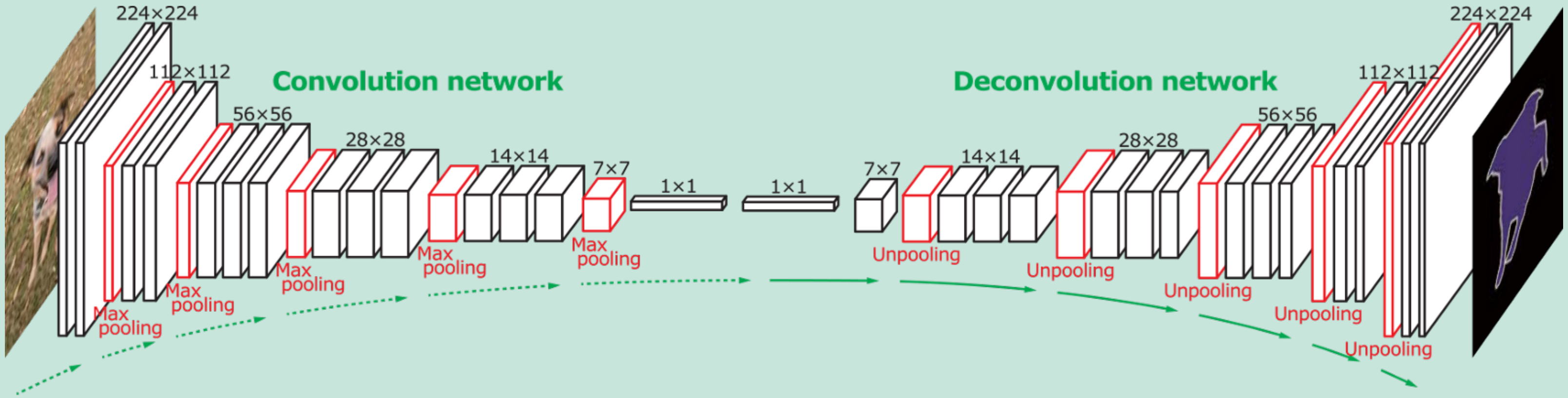
- Automized?



State of the art

CNNs (1/2)

- Specialized in detecting patterns
- Encoder – decoder structure for semantic segmentation

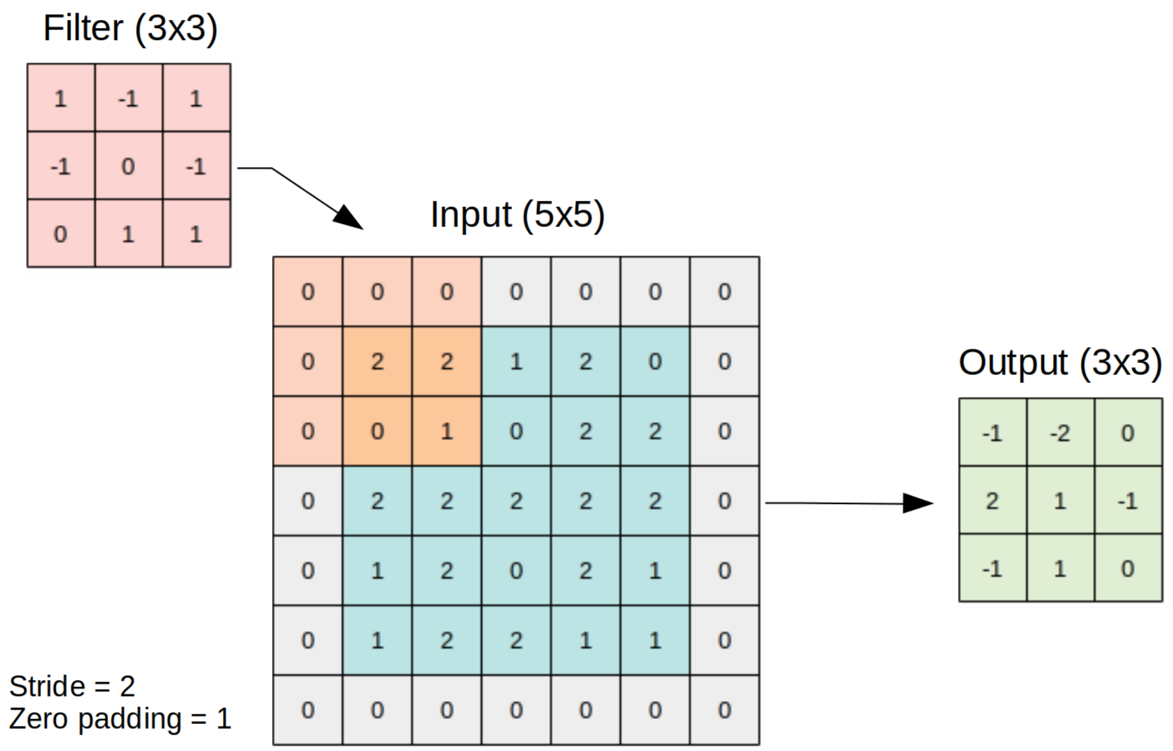


Source: Noh, Hong & Han (2015)

CNNs (2/2)

Layer types:

- Convolutional
- Transposed-convolutional
- Non-linear function
- Spatial pooling



Related work

Added value of 2.5D or 3D

- *Couprie et al. (2013)*
  - RGB-D indoor scene segmentation
  - Addition of **depth** increases labeling precision!

Semantic segmentation of aerial imagery + height

- *Kampffmeyer et al. (2016)* & *Liu et al. (2017)*
  - No examination of **added value** of height info
  - No examination of most suitable **height type**

Data stacking versus data fusion

- *Hazirbas et al. (2017)*
  - **Fusion outperforms stacking** approaches for indoor scenes with depth information

Gaps in research

Added value of height information for semantic segmentation of **aerial imagery?**

Does data **fusion** or **data stacking** work better for semantic segmentation of aerial imagery?

What **type of height information** can best be presented to the network?

Objectives

Objectives

1. Generate a **CNN model** that performs **automatic, pixel-level semantic segmentation** of remotely sensed imagery.
2. Examine the **added value** of the included height information for the semantic segmentation of aerial imagery.
3. Explore in **what way** the height information can best be **presented** to the algorithms.

Research questions

Research question

To what extent can **convolutional neural networks** be used for **automatic** semantic segmentation of RGB-Z aerial imagery?

Sub-questions

Which neural network **architectures** are a suitable **starting point** for semantic segmentation of aerial RGB-Z imagery?

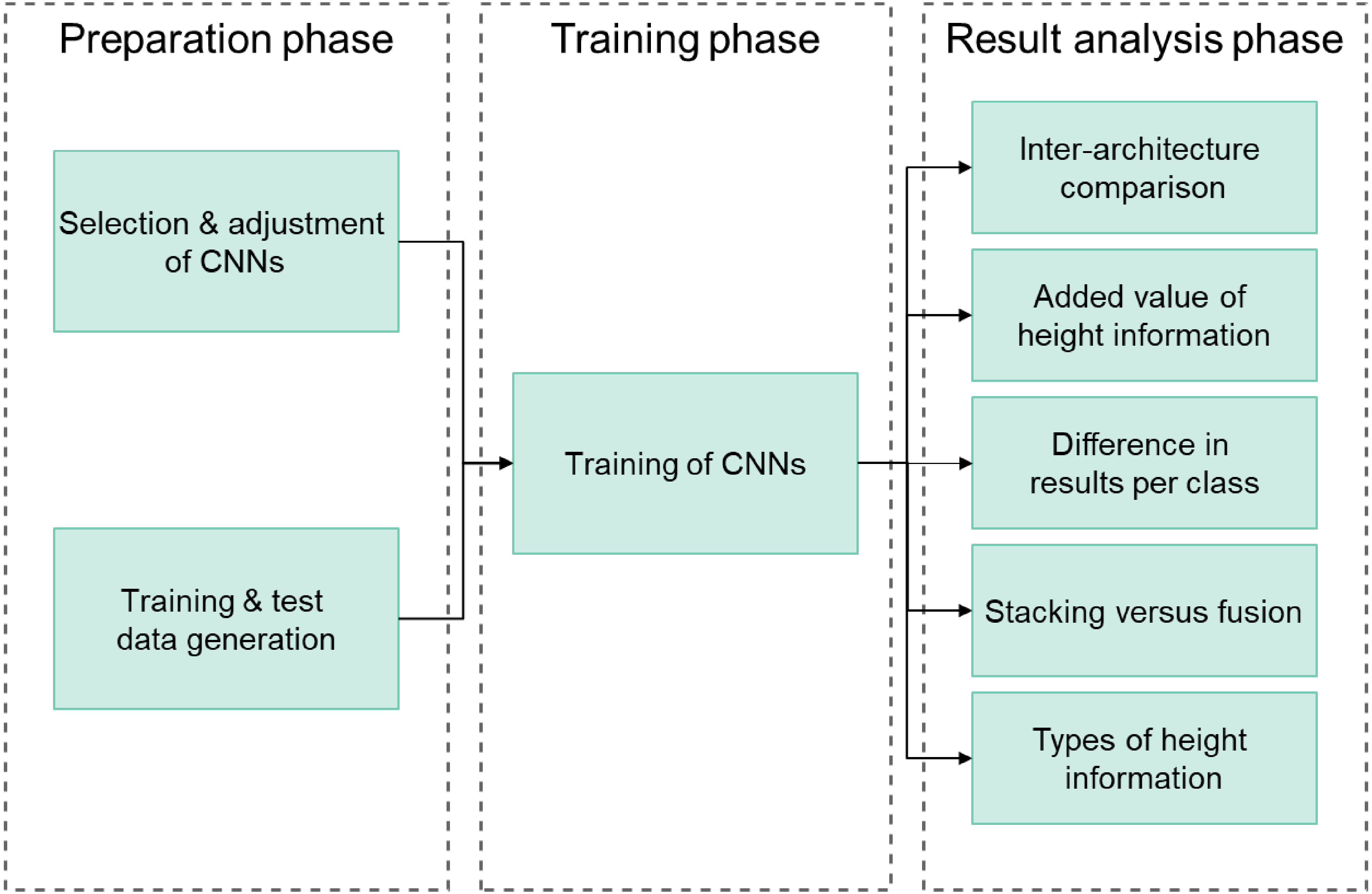
To what extent does the **addition of height information** improve semantic segmentation results?

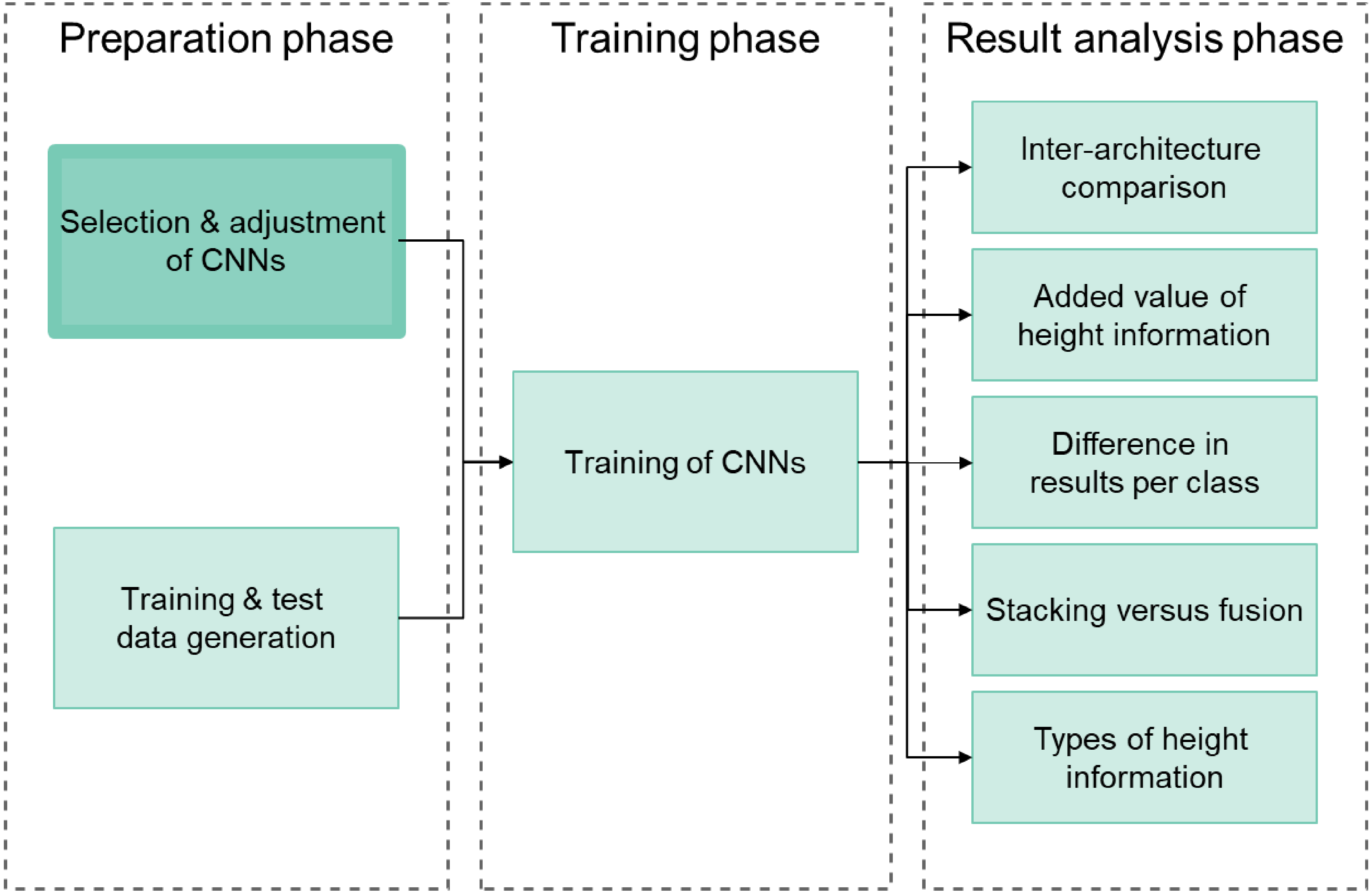
For which **classes** is the segmentation most successful; for *building*, *road*, *water* or *other*?

How does the performance compare of different approaches on **combining height information with RGB** information (*stacking* and *fusion*) in a network?

What **type of height** information provided to a network leads to the most accurate results?

Methodology


Preparation phase
Training phase
Result analysis phase
Selection & adjustment of CNNs
Training & test data generation
Training of CNNs
Inter-architecture comparison
Added value of height information
Difference in results per class
Stacking versus fusion
Types of height information


Preparation phase
Training phase
Result analysis phase
Selection & adjustment of CNNs
Training & test data generation
Training of CNNs
Inter-architecture comparison
Added value of height information
Difference in results per class
Stacking versus fusion
Types of height information

Selection of CNNs

Suitable when adherent to criteria:

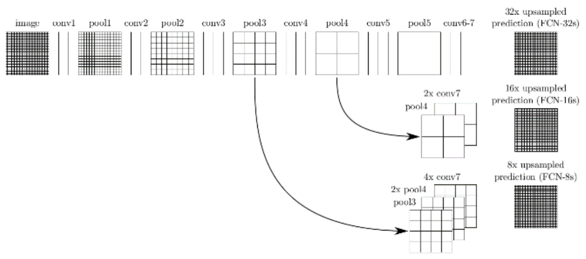
- **Successful performance** on any type of imagery
- Source **code available**, no license restrictions
- Not specific to one task & allows for input **own data**
- Implementation in **Python**

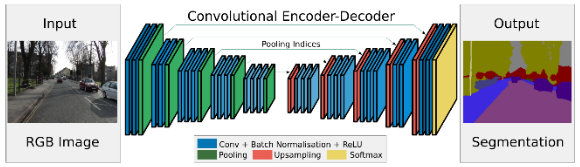
Led to selection of 4 architectures:

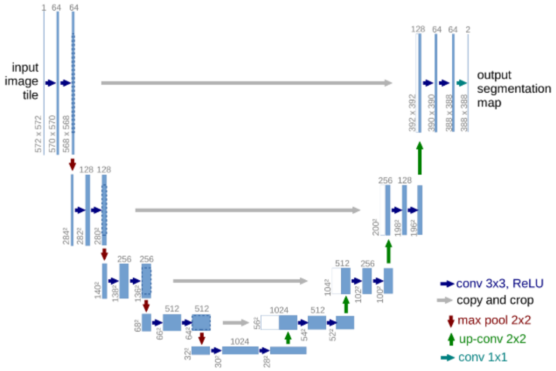
- FCN-8s
- SegNet
- U-Net
- FuseNet-SF5

Architectures

Data stacking





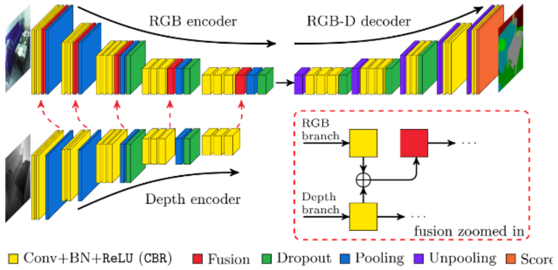


FCN-8S
- Learns to deconvolve input feature maps
- Focusses on details

SegNet
- Preserves high-frequency information
- Focusses on boundaries

U-Net
- Preserves neighboring information
- Focusses on limited training data

Data fusion



FuseNet-SF5
- Two encoders
- Allows for learning more distinct features

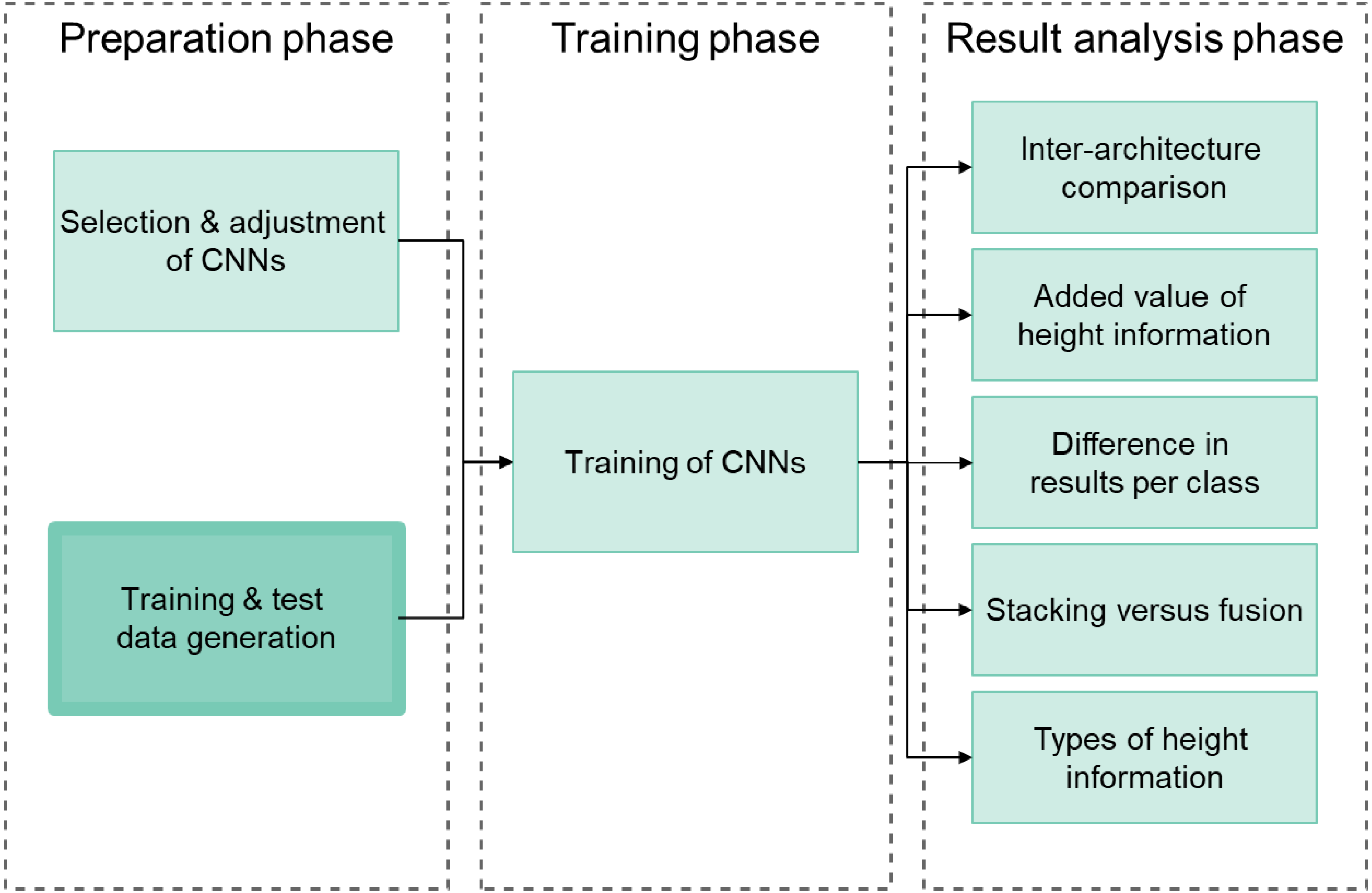
Sources LTR: Long et al. (2015), Badrinarayanan et al. (2017), Ronneberger et al. (2015), Hazirbas et al. (2017)

Architecture implementations

Python

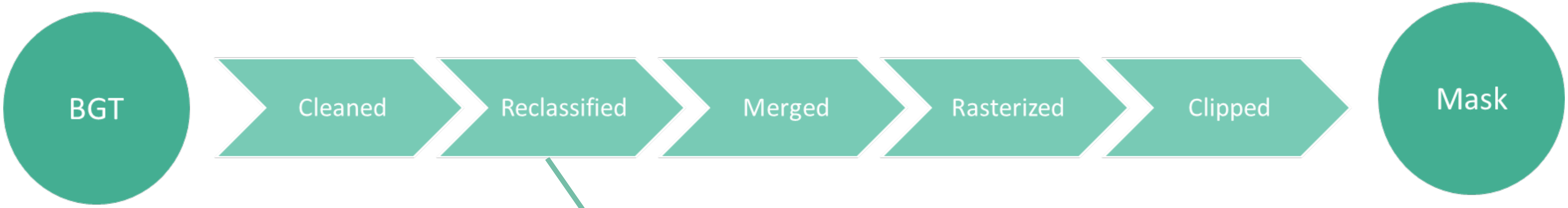
PyTorch

PyTorch-SemSeg repository


Preparation phase
Training phase
Result analysis phase
Selection & adjustment of CNNs
Training & test data generation
Training of CNNs
Inter-architecture comparison
Added value of height information
Difference in results per class
Stacking versus fusion
Types of height information

Training & test data generation

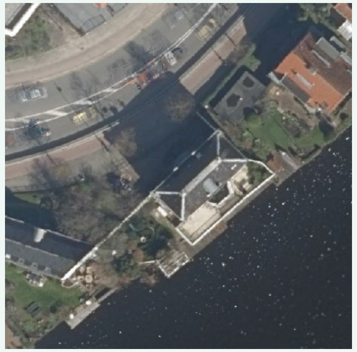
Preparing the BGT



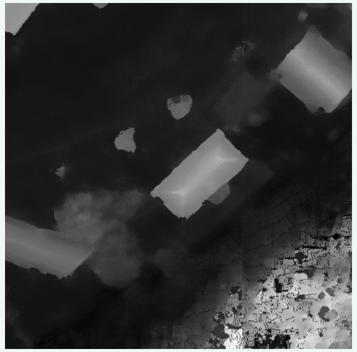
Class	BGT	
Building	Gebouw installatie	Pand
	Overig bouwwerk	
Road	Overbruggingsdeel	Wegdeel
Water	Waterdeel	
Other	Begroeid terreindeel	Ondersteunend waterdeel
	Gebouwinstallatie	Ondersteunend wegdeel
	Kunstwerkdeel	Openbare ruimte
	Obegroeid terreindeel	Overig bouwwerk

Training & validation data generation

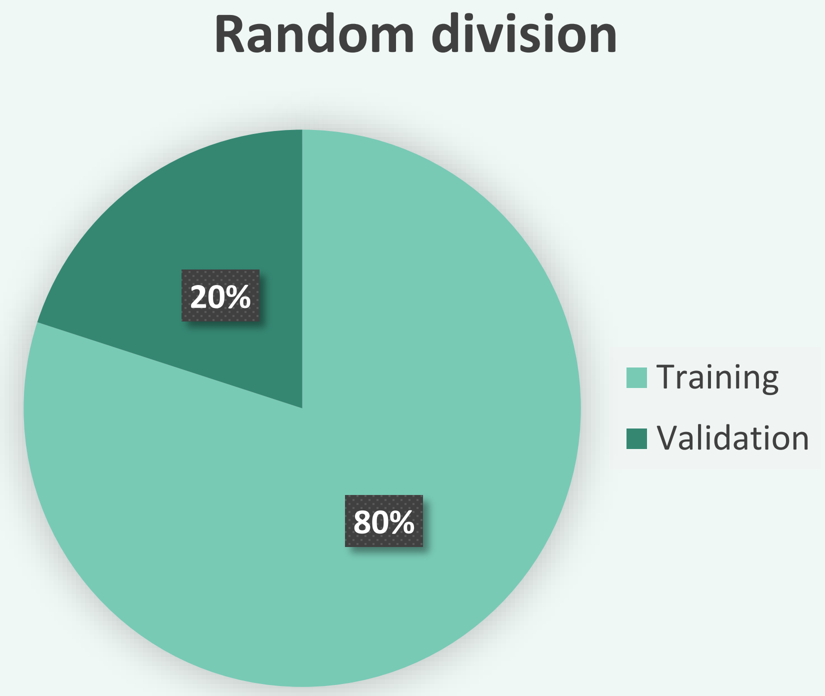
Imagery:
- True ortho (READAR)
- Corrected for relief displacement
- 1600 tiles, 512x512 pixels per tile
- Every pixel 10x10 cm



Height information:
- DSM (READAR)
- Matching to true ortho



Mask layer:
- Cleaned & rasterized BGT:
  - 1 class label per pixel



Height approaches

Absolute

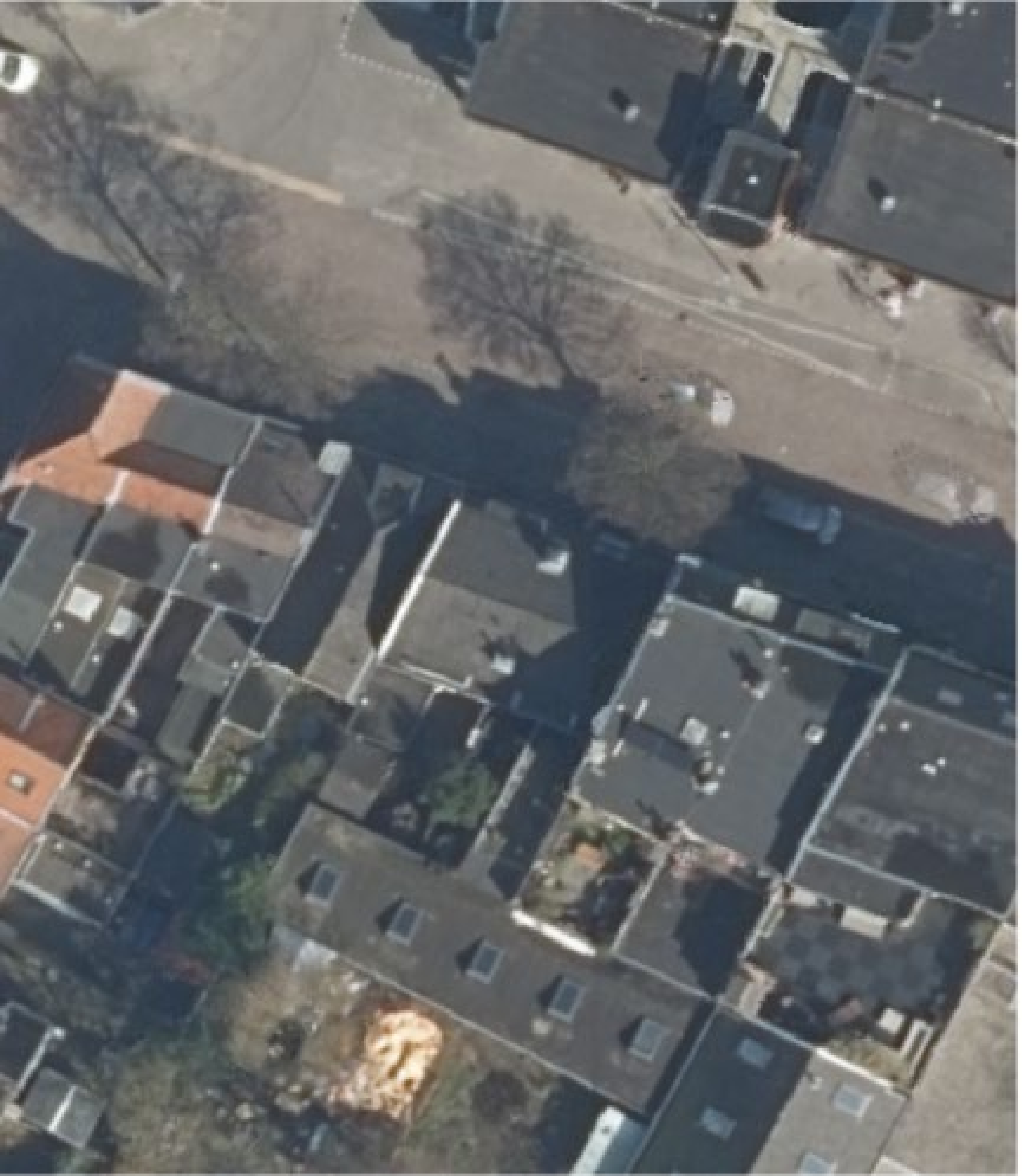
- DSM

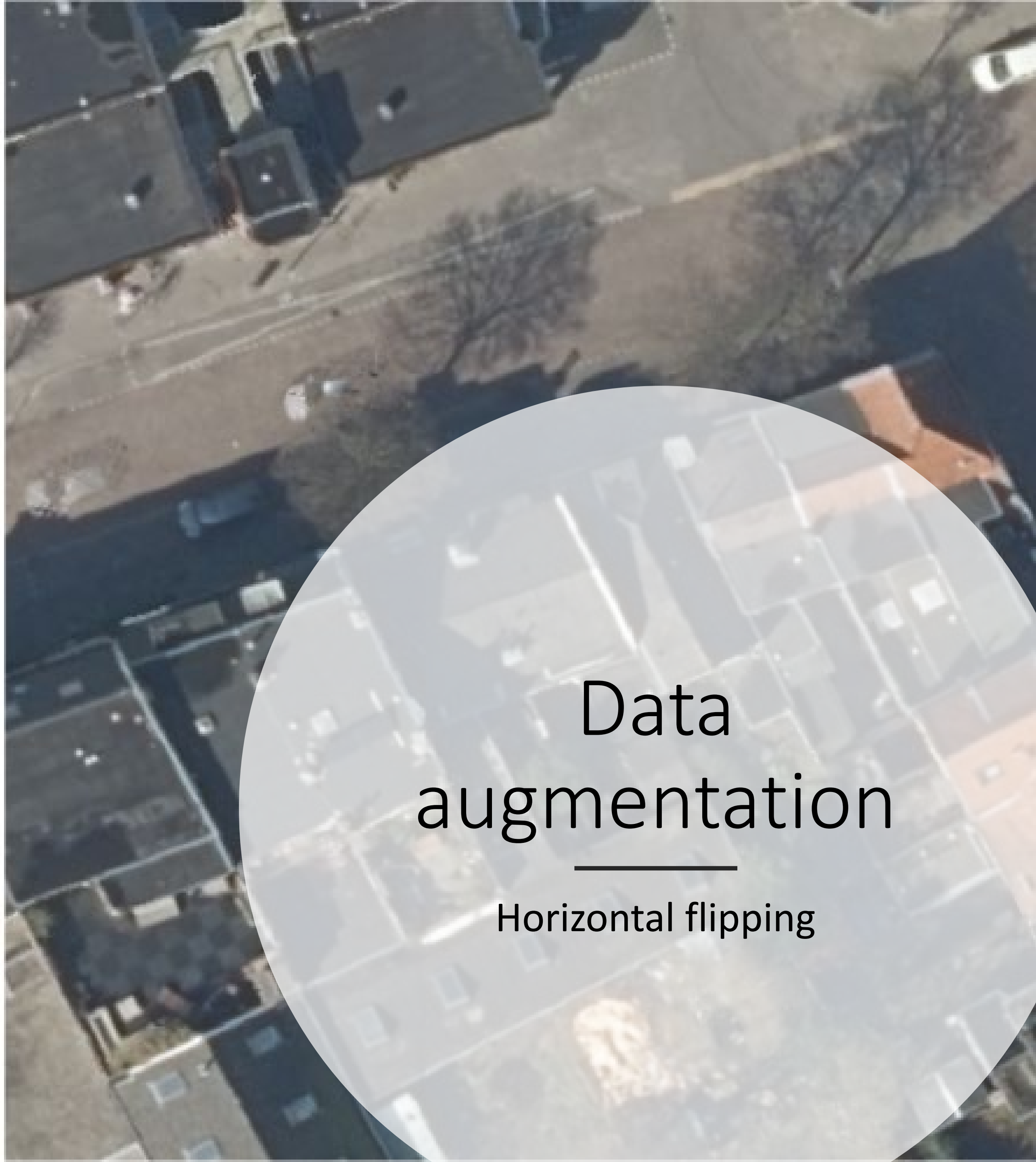
Rescaled

- Min-max feature scaling [0-1]
  - Tile-level
  - Whole train/test area
  - $X' = \frac{X - Xmin}{Xmax - Xmin}$

Relative

- DSM-DTM
  - Pixel-level
  - Tile-level
- DTM from AHN3 (0.5m)

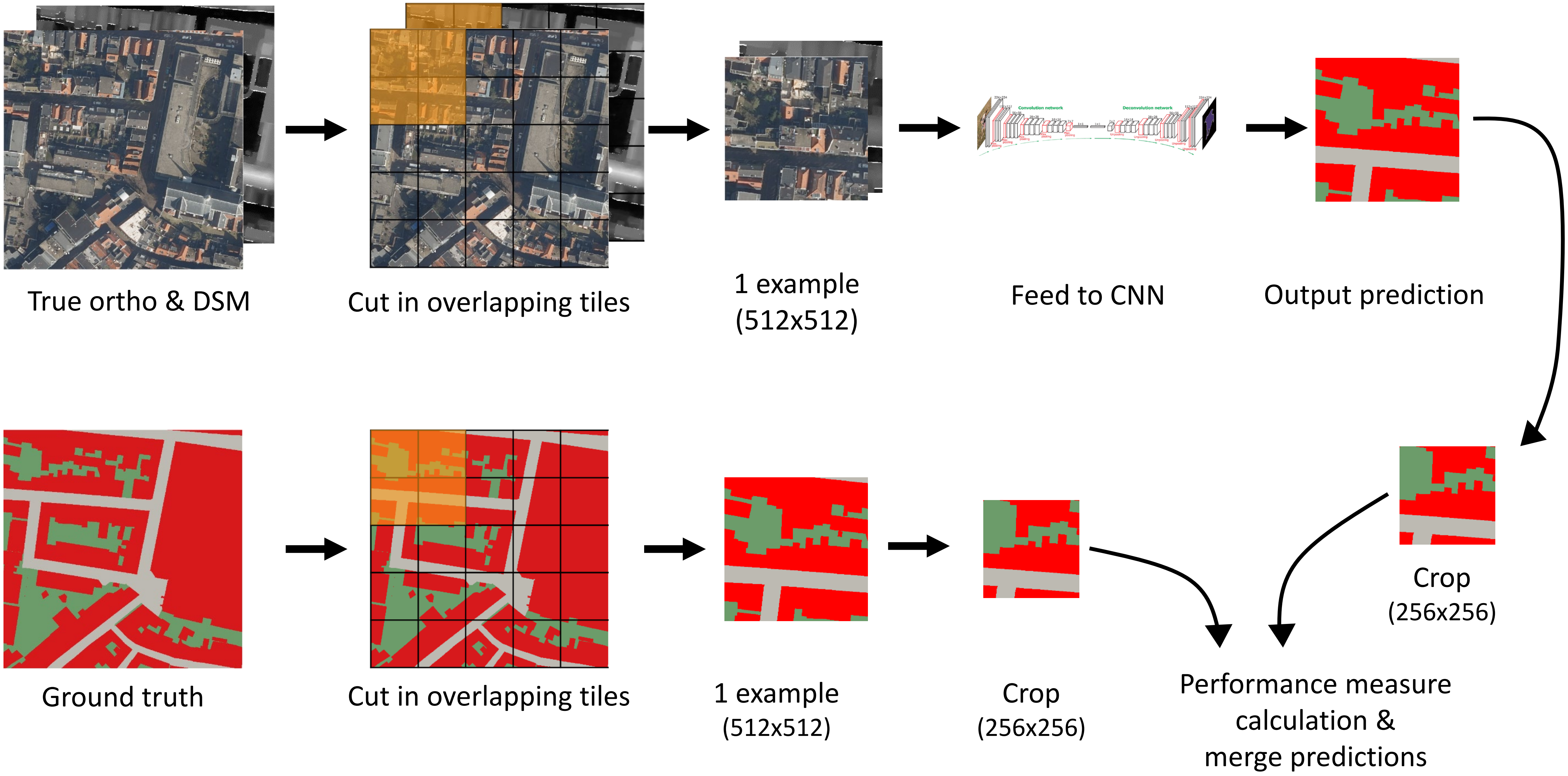


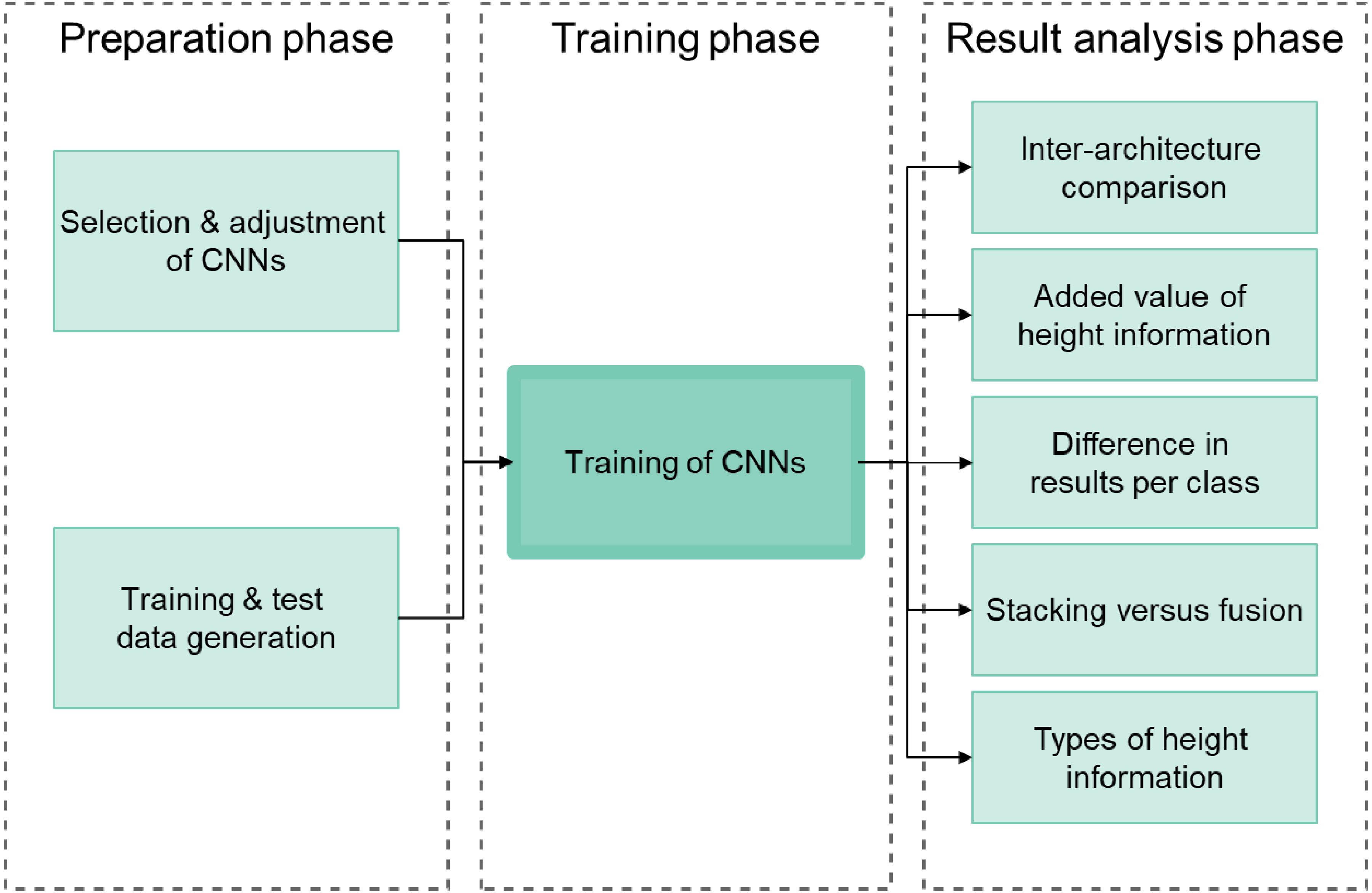


Data augmentation

Horizontal flipping

Test data and inference


True ortho & DSM
Cut in overlapping tiles
1 example
(512x512)
Feed to CNN
Output prediction
Crop
(256x256)
Ground truth
Cut in overlapping tiles
1 example
(512x512)
Crop
(256x256)
Performance measure
calculation &
merge predictions


Preparation phase
Training phase
Result analysis phase
Selection & adjustment of CNNs
Training & test data generation
Training of CNNs
Inter-architecture comparison
Added value of height information
Difference in results per class
Stacking versus fusion
Types of height information

Training of CNNs

- External server
- Performance measures

$$F1_i = 2\frac{precision_i \times recall_i}{precision_i + recall_i}$$

$$precision = \frac{p_{ii}}{C_i}, recall = \frac{p_{ii}}{P_i}$$

$$mIoU = \frac{1}{k+1}\sum_{i=0}^{k}\frac{p_{ii}}{\sum_{j=0}^{k} p_{ij} + \sum_{j=0}^{k} p_{ji} - p_{ii}}$$

k = number of classes
j = predicted class of pixel
p_{ij} = number of false positives
Pi = number of pixels assigned to class i by prediction
Ci = actual total number of pixels belonging to class i
i = actual class of pixel
p_{ii} = number of true positives
p_{ji} = number of false negatives

Experimental setup

Optimize on RGB (no height)

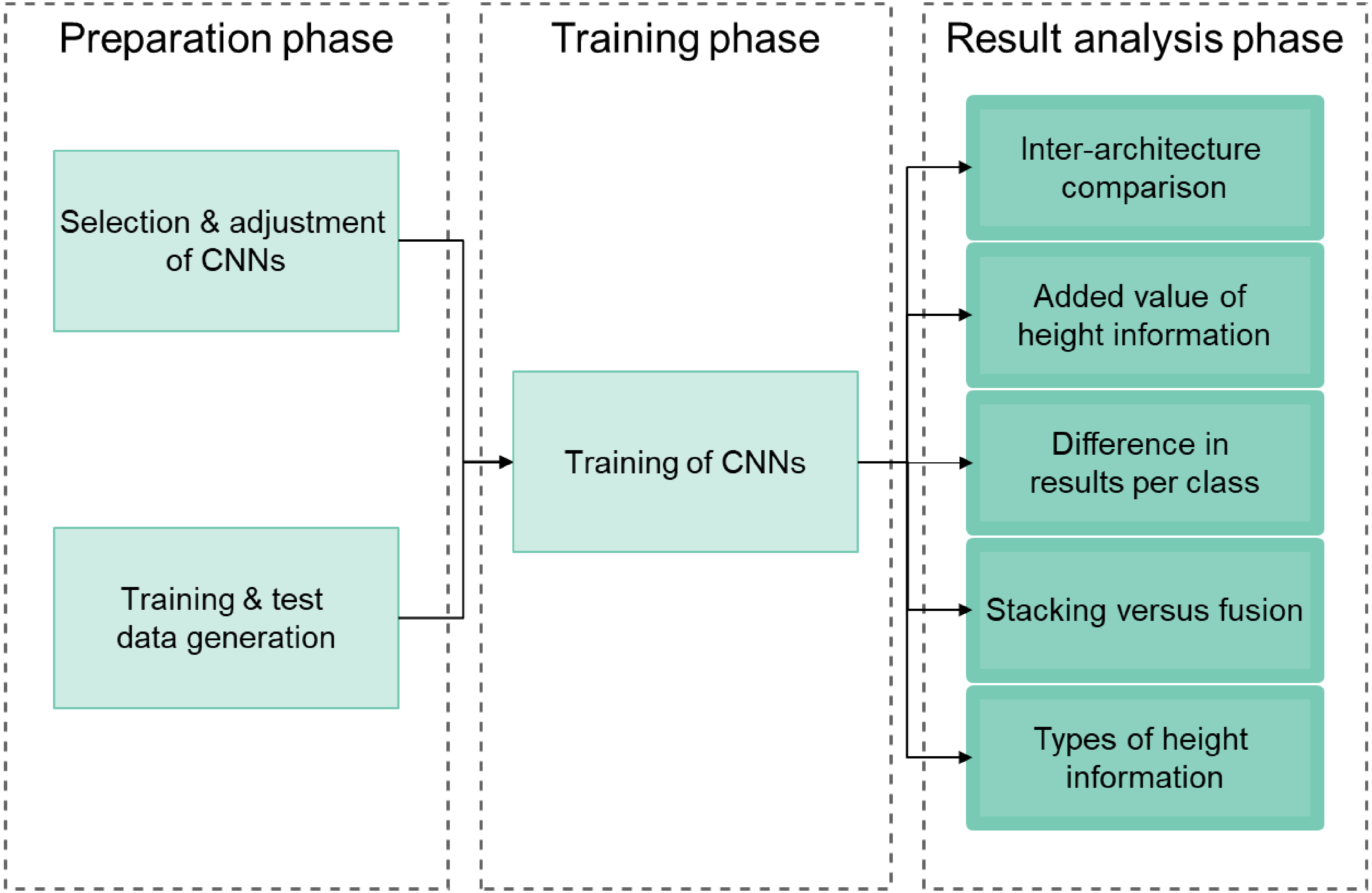
Train on RGB-Z (with height)

FuseNet-SF5

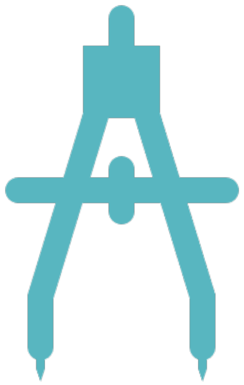
Height approaches

		RGB			*RGB-Z (data stacking)*			*RGB-Z (data fusion)*
Hyperparameter	**Options**	**FCN-8s**	**SegNet**	**U-Net**	**FCN-8s**	**SegNet**	**U-Net**	**FuseNet-SF5**
Weight initialization	Pretrained / random	x	x		x	x		x
(Initial) learning rate	1e-3 / 1e-4 / 1e-5	x	x	x				x
Optimizer	SGD / Adam	x	x	x				x
Loss function	CP / WCP	x	x	x				x
# epochs no improvement	10 / 20 / 50	x	x	x				x
Horizontal flipping	Yes/no	x	x	x				x
Height type	AH / SHT / SHW / RHP / RHT				AH & SHT	AH & SHT	AH & SHT	x

CP = cross-entropy, WCP = weighted cross-entropy, AH = Absolute height, SHT = Rescaled height [0-1] (tile-level), SHW = Rescaled height [0-1] (whole area), RHP = Relative height (pixel-level), RHT = Relative height (tile-level)


Preparation phase
Training phase
Result analysis phase
Selection & adjustment of CNNs
Training & test data generation
Training of CNNs
Inter-architecture comparison
Added value of height information
Difference in results per class
Stacking versus fusion
Types of height information

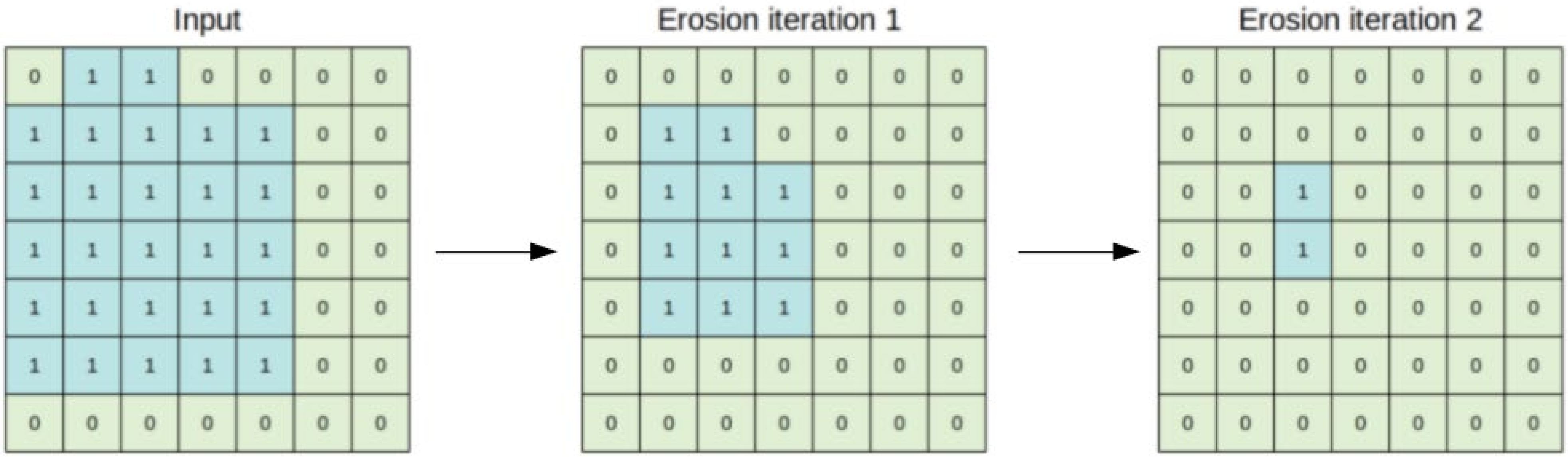
Drawing conclusions



(m)IoU

Visual

Error maps and morphological erosion



Object-level performance

Detection of ground truth objects

- Percentage of **correctly classified** pixel **per object** in ground truth

False positives?

- **Polygonize** eroded false-positive error maps

Results & analysis

Hyperparameters

Hyperparameter	FCN-8s	SegNet	U-Net	FuseNet-SF5
Weight initialization	Pretrained	Pretrained	Random	Pretrained
(Initial) learning rate	1e-4	1e-4	1e-4	1e-4
Optimizer	Adam	Adam	Adam	Adam
Loss function	CP	CP	CP	CP
# epochs no improvement	50	50	50	50
Horizontal flipping	Yes	Yes	Yes	Yes
Height type (only with RGB-Z)	SHT	SHT	SHT	RHP

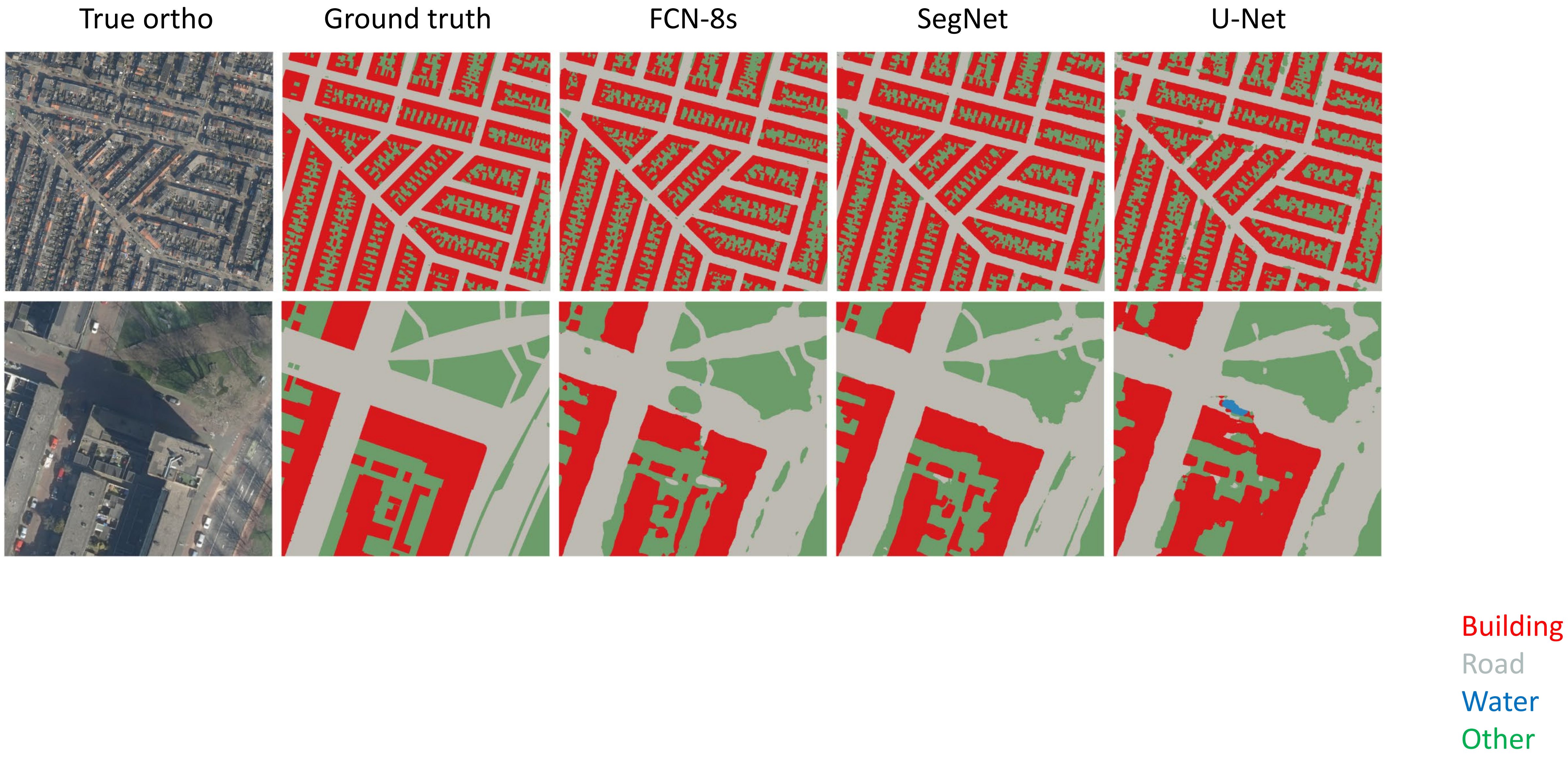
- CP = Cross-entropy
- AH = Absolute height
- SHT = Rescaled height [0-1] (tile-level)
- RHP = Relative height (pixel-level)

RGB baseline comparison

Model	mIoU	F1
FCN-8s	0.8121	0.8958
SegNet	**0.8219**	**0.9015**
U-Net	0.7637	0.8647

Performance measures on test data


True ortho
Ground truth
FCN-8s
SegNet
U-Net
Building
Road
Water
Other


True ortho
Ground truth
FCN-8s
SegNet
U-Net
Building
Road
Water
Other

Data stacking: RGB vs. RGB-Z

Overall performance

Model	Input	mIoU	F1
FCN-8s	RGB	0.8121	0.8958
FCN-8s	RGB-Z	0.8177	0.8990
SegNet	RGB	0.8129	0.9015
SegNet	RGB-Z	**0.8257**	**0.9039**
U-Net	RGB	0.7637	0.8647
U-Net	RGB-Z	0.7851	0.8786

Performance measures on test data

True ortho | DSM | Ground truth | FCN-8s (RGB) | FCN-8s (RGB-Z)

Building
Road
Water
Other

Data stacking: RGB vs. RGB-Z

Class performance

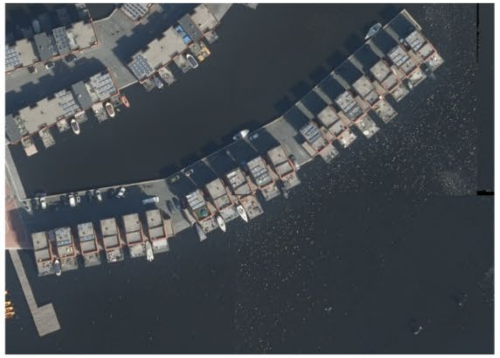
Model	Input	Building	Road	Water	Other
FCN-8s	RGB	0.8305	0.7822	0.8661	0.7698
FCN-8s	RGB-Z	**0.8567**	0.7714	0.8700	0.7725
		+0.0262	-0.0108	+0.0039	+0.0027
SegNet	RGB	0.8426	0.7810	**0.8907**	0.7735
SegNet	RGB-Z	0.8538	**0.7827**	0.8841	**0.7822**
		+0.0112	+0.0017	-0.0066	+0.0087
U-Net	RGB	0.7814	0.6974	0.8353	0.7225
U-Net	RGB-Z	0.8384	0.7134	0.8365	0.7521
		+0.0570	+0.0160	-0.0170	+0.0296

Performance measures on test data

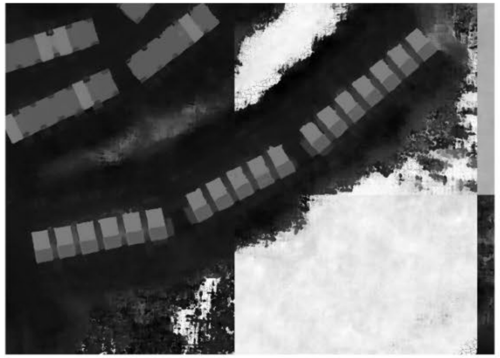
Stacking vs. fusion

Model	Building	Road	Water	Other	mIoU
SegNet (RGB-Z)	0.8538	**0.7827**	0.8841	0.7822	0.8257
FuseNet-SF5	**0.8723**	0.7767	**0.9143**	**0.7890**	**0.8381**
	+0.0185	-0.0060	+0.0302	+0.0068	+0.0124

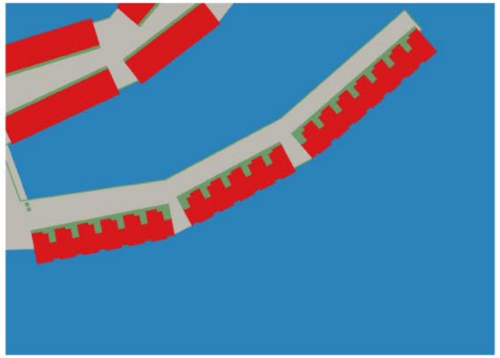
Performance measures on test data



True ortho



DSM



Ground truth

SegNet (RGB-Z)

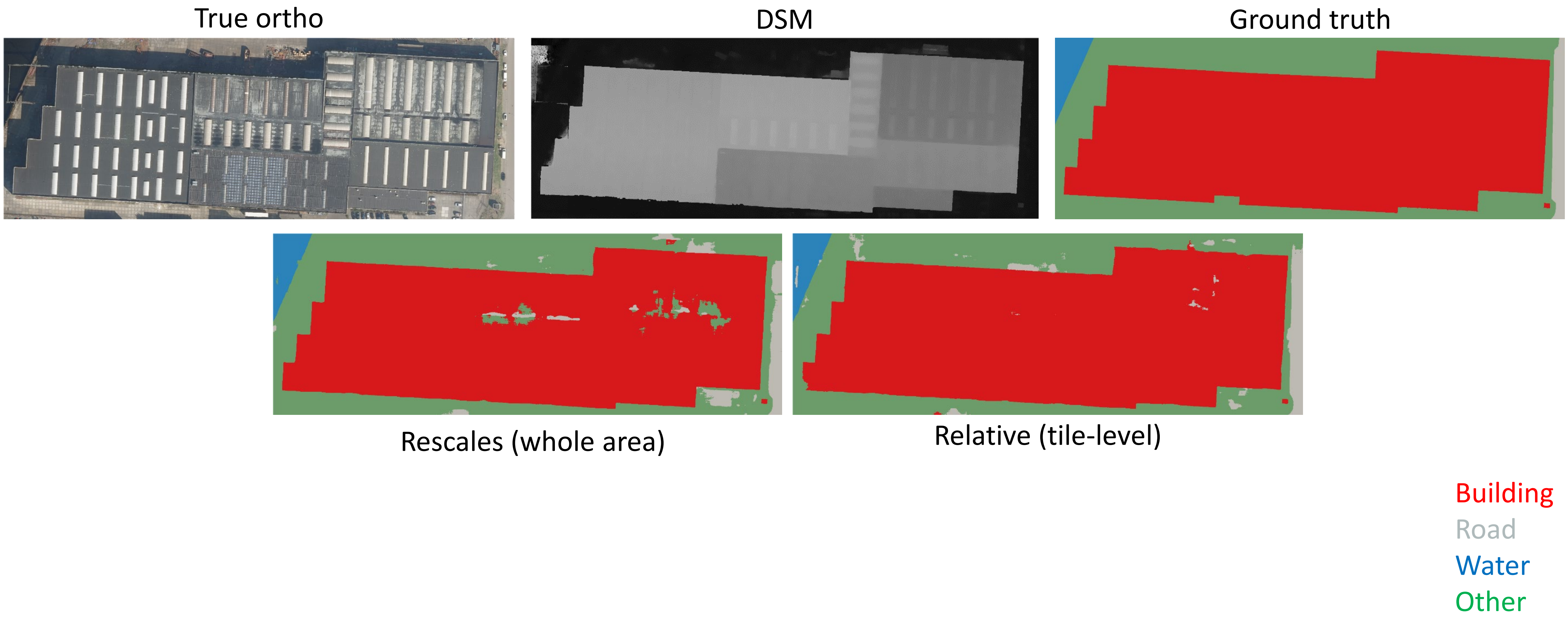
FuseNet-SF5

Building
Road
Water
Other

Height approaches

Height type	Building	Road	Water	Other	mIoU
Absolute	0.8723	0.7767	0.9143	0.7890	0.8381
Rescaled [0-1] (tile-level)	0.8671	0.7750	0.9023	0.7860	0.8326
Rescaled [0-1] (whole area)	0.8708	0.7846	**0.9152**	0.7897	0.8401
Relative (pixel-level)	0.8744	**0.7865**	0.9131	**0.7966**	**0.8427**
Relative (tile-level)	**0.8792**	0.7785	0.9070	0.7891	0.8384

IoU performance on the test data of FuseNet-SF5


True ortho
DSM
Ground truth
Rescales (whole area)
Relative (tile-level)
Building
Road
Water
Other

True ortho

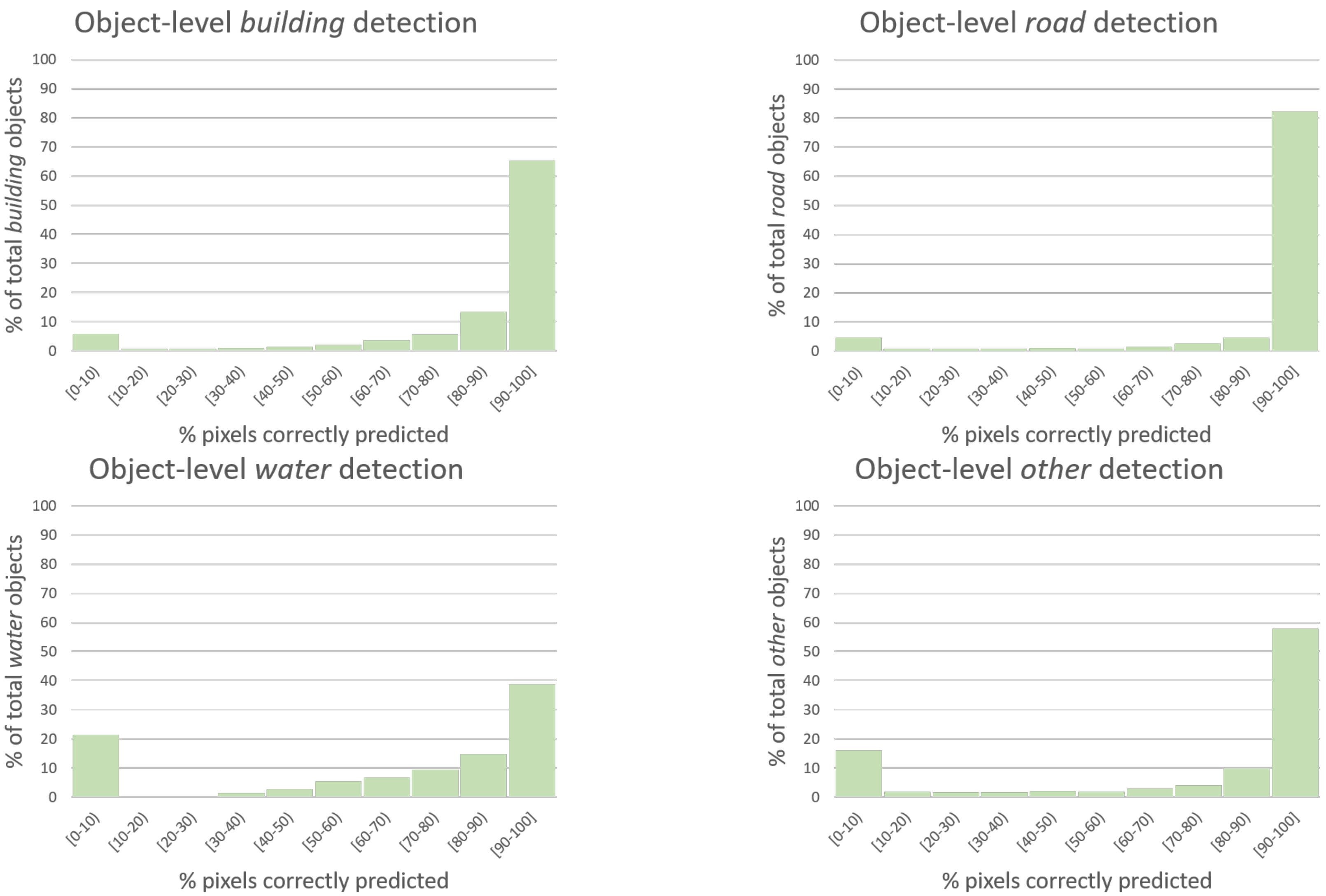
DSM

Ground truth

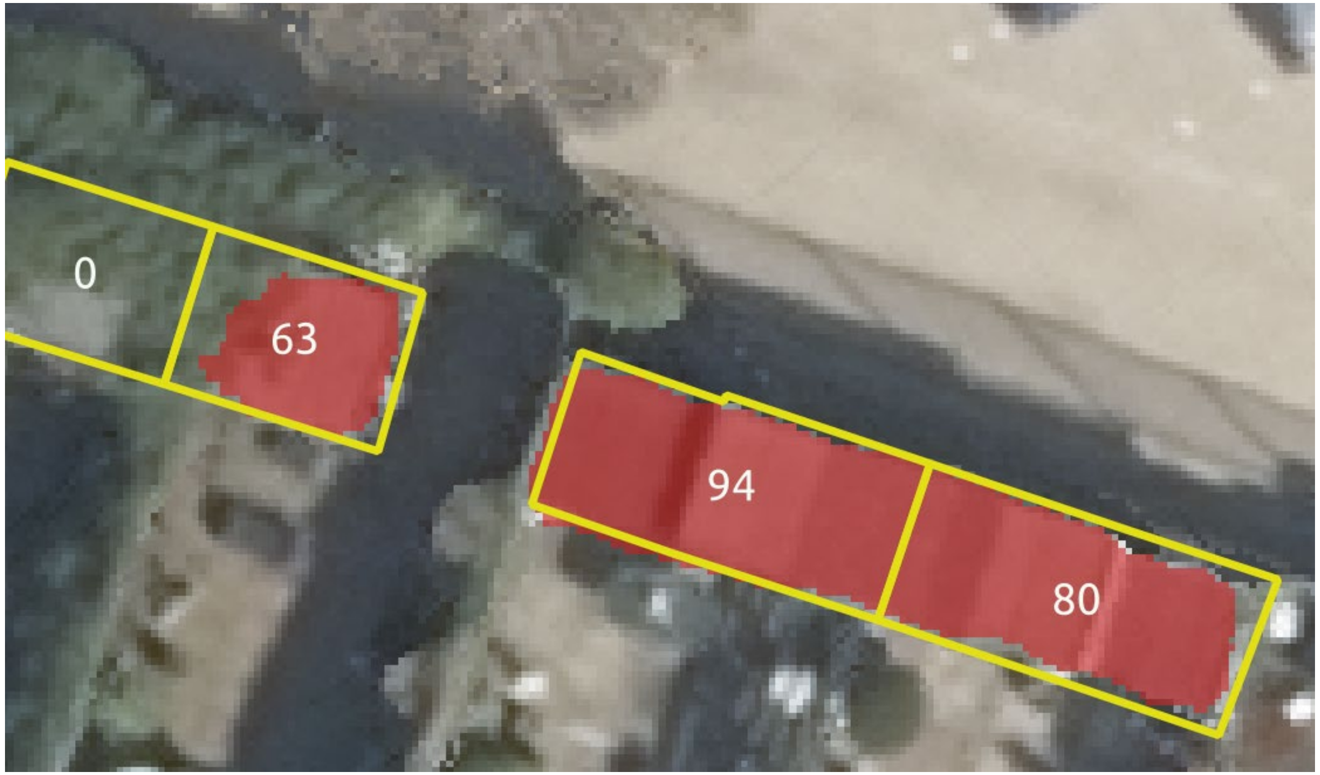
FuseNet-SF5
relative height
(pixel-level)

Building
Road
Water
Other

Object-level detection



Missed objects: *Building*



- Limited visibility due to **trees**
- Error in **BGT**
- Error of **algorithm** (rare)

Missed objects: *Road*

- Limited visibility due to trees
- Error in BGT
- Error of algorithm
- **Shade**

Missed objects: *Water*

- Limited visibility due to **trees**
- Thin water bodies (**ditches**)

Missed objects: *Other*

- **Small objects** that are not clearly distinctive
- Thin segments **misinterpreted** for **road**
- Errors in **BGT**

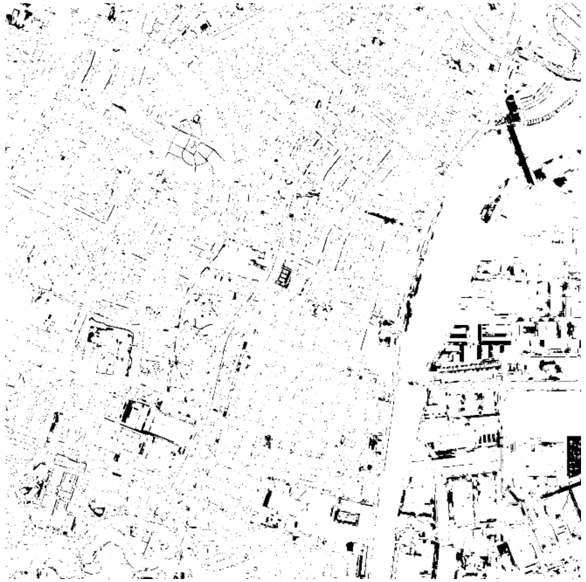
Object-level false positives

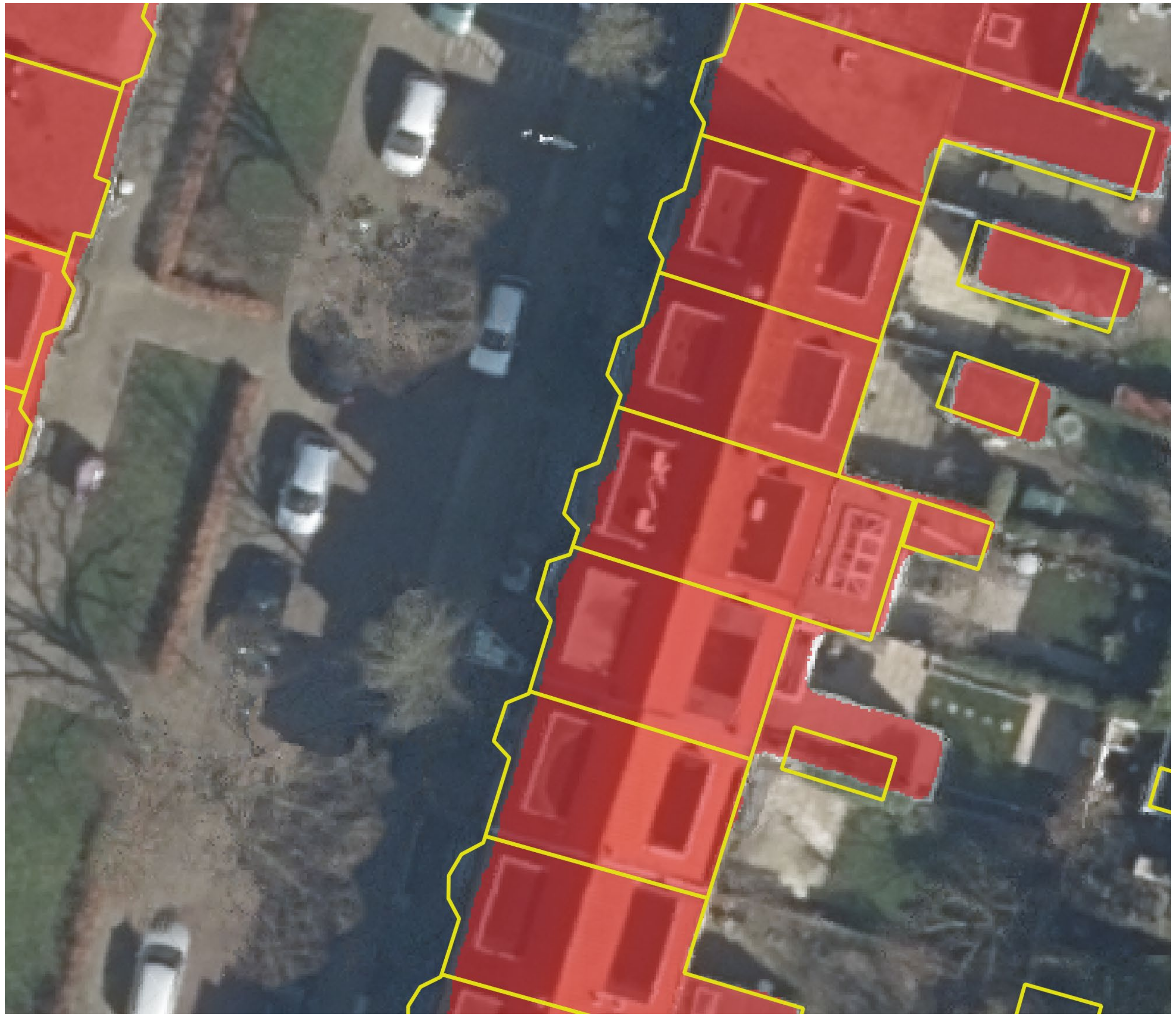
Red =	False positive polygons for *building*
Yellow =	Ground truth for *building*

Disputable inconsistencies

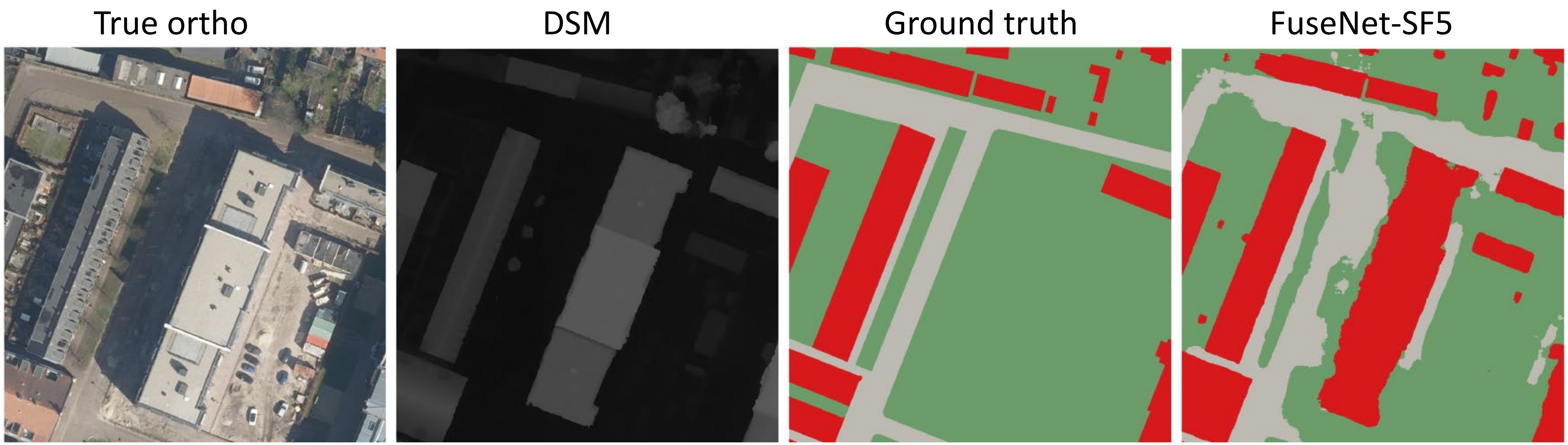
Not eroded

Eroded





Misplaced objects in BGT (yellow) are correctly detected by algorithm (red)



Building
Road
Water
Other

Discussion

Methodology limitations

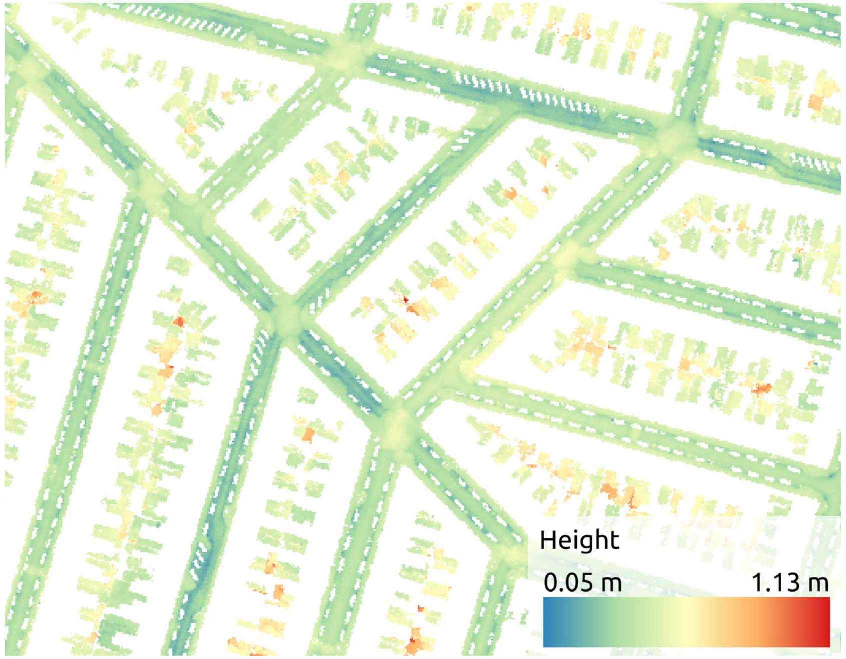
Significance?

“Pixel”-level subtraction?

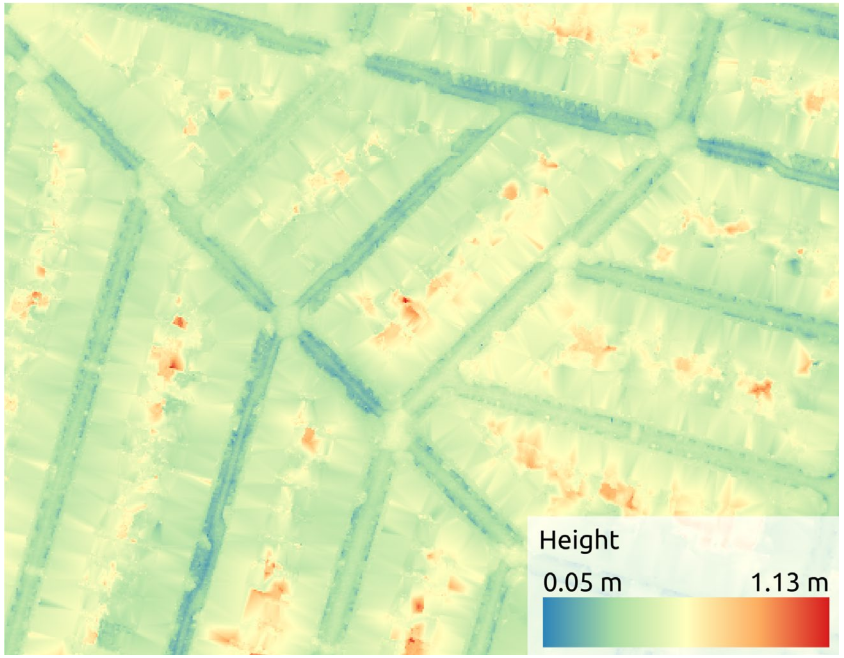
Influence interpolated holes in DTM?

Influence interpolated holes DTM?

True ortho



DTM



Interpolated DTM

Conclusions

Conclusions

To what extent can convolutional neural networks be used for automatic semantic segmentation of RGB-Z aerial imagery?

Which neural network architectures are a suitable starting point for semantic segmentation of aerial RGB-Z imagery?

FCN-8s, SegNet, U-Net, FuseNet-SF5

- *Showed successful semantic segmentation*
- *Openly available implementation*
- *Allowed for use of own data*

To what extent does the addition of height information improve semantic segmentation results?

- *On average performance improved by 1% (mIoU)*
- *Valuable and essential information is encoded in height data*

For which classes is the segmentation most successful; for building, road, water or other?

- *Most successful for 'water' and 'building'*
- *'Building' benefits most from addition of height information*
- *Best performing algorithm detected in the ground truth over 90% of:*
  - *65% of 'building' objects*
  - *82% of 'road' objects*
  - *58% of 'other' objects*
  - *39% of 'water' objects*

How does the performance compare of different approaches on combining height information with RGB information (*stacking* and *fusion*) in a network?

- *Fusion outperforms stacking*
- *Fusion allows for different types of features learned from height*
- *Fusion exploits potential of height information to a higher degree*

What type of height information provided to a network leads to the most accurate results?

- *Relative height outperforms absolute height*
- *Pixel-level, relative height shows higher mIoU than tile-level relative height*
- *Part of success probably due to flat nature of Haarlem*

Contributions

Height information can **add value** to semantic segmentation of aerial RGB imagery

Adding height information through **data fusion** can result in higher segmentation quality of **aerial imagery** than when data stacking is used

Providing **relative height**, rather than absolute height, to a network can improve semantic segmentation quality of **aerial imagery**, especially for large objects

Future work

Future work

- BGT error removal
- Relative height without DTM of AHN
- Fusing stacked height information

Thank you for your attention!

Amber E. Mulder

TU Delft

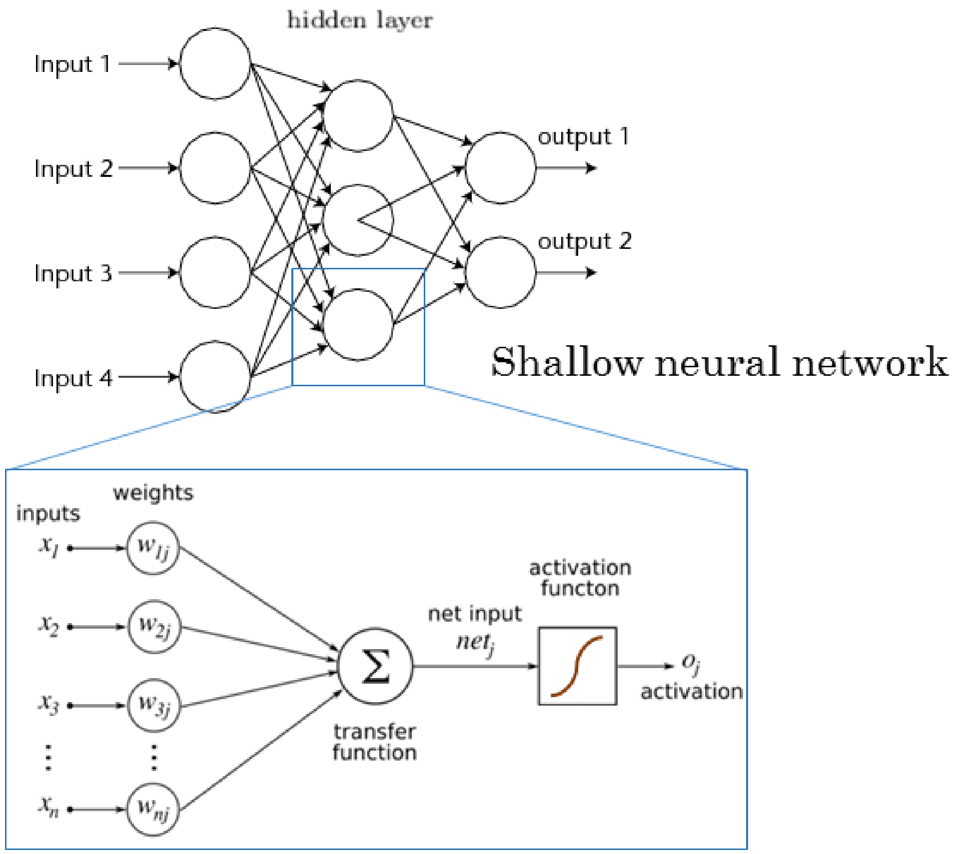
READAR
real estate radar

References

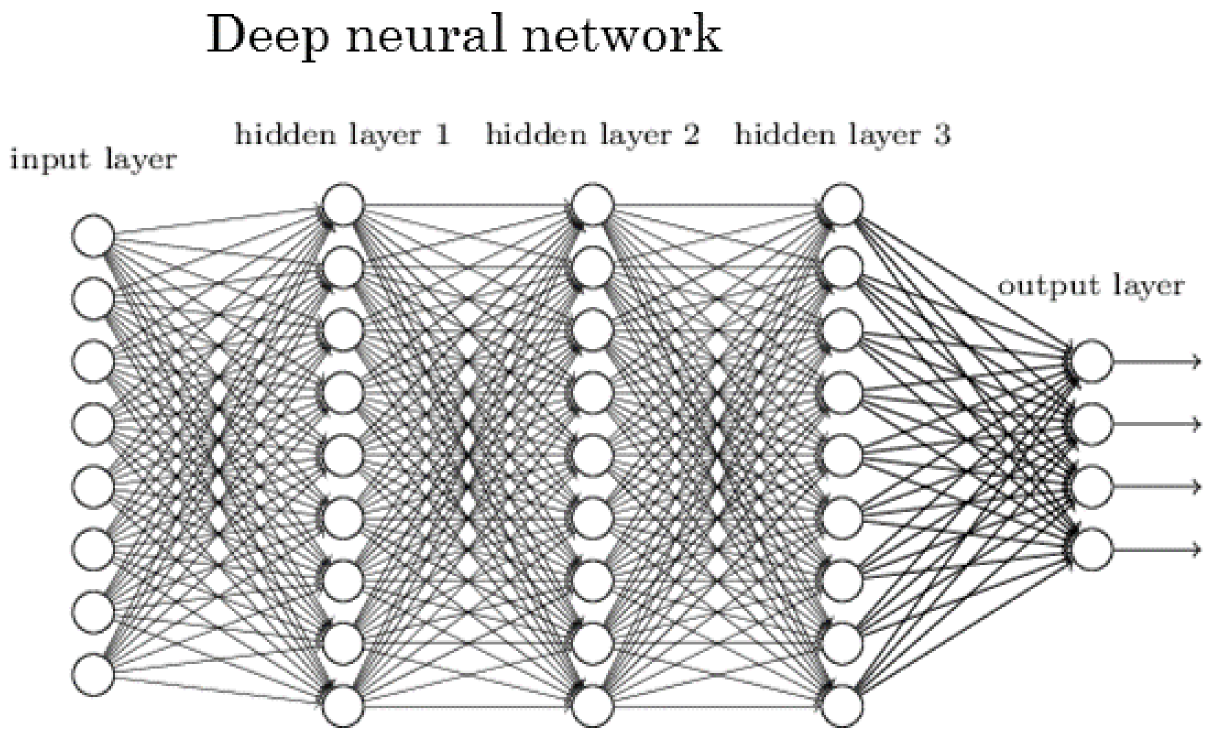
- Audebert, N., Le Saux, B., and Lef`evre, S. (2018). Beyond RGB: Very high resolution urban remote sensing with multimodal deep networks. ISPRS Journal of Photogrammetry and Remote Sensing, 140:20–32.
- Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE transactions on pattern analysis and machine intelligence, 39(12):2481–2495.
- Couprie, C., Farabet, C., Najman, L., and LeCun, Y. (2013). Indoor Semantic Segmentation using depth information.
- Hazirbas, C., Ma, L., Domokos, C., and Cremers, D. (2017). FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-Based CNN Architecture. In Lai, S.-H., Lepetit, V., Nishino, K., and Sato, Y., editors, Computer Vision – ACCV 2016, volume 10111, pages 213–228. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.
- Kampffmeyer, M., Salberg, A.-B., and Jenssen, R. (2016). Semantic Segmentation of Small Objects and Modeling of Uncertainty in Urban Remote Sensing Images Using Deep Convolutional Neural Networks. pages 1–9.
- Liu, Y., Minh Nguyen, D., Deligiannis, N., Ding, W., and Munteanu, A. (2017). Hourglass-Shape Network Based Semantic Segmentation for High Resolution Aerial Imagery. Remote Sensing, 9(6):522.
- Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3431–3440.
- Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision* (pp. 1520-1528).
- Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation.
- RSIP Vision (n.d.). Deep Learning and Convolutional Neural Networks: RSIP Vision Blogs. https://www.rsipvision.com/exploring-deep-learning/. Accessed 4 Apr. 2020.
- SUMMER_story (n.d.). Learning Tensorflow. https://summer-story.tistory.com/6. Accessed 4 Apr. 2020.

Extra slides

Deep learning



Source: RSIP Vision (n.d.)



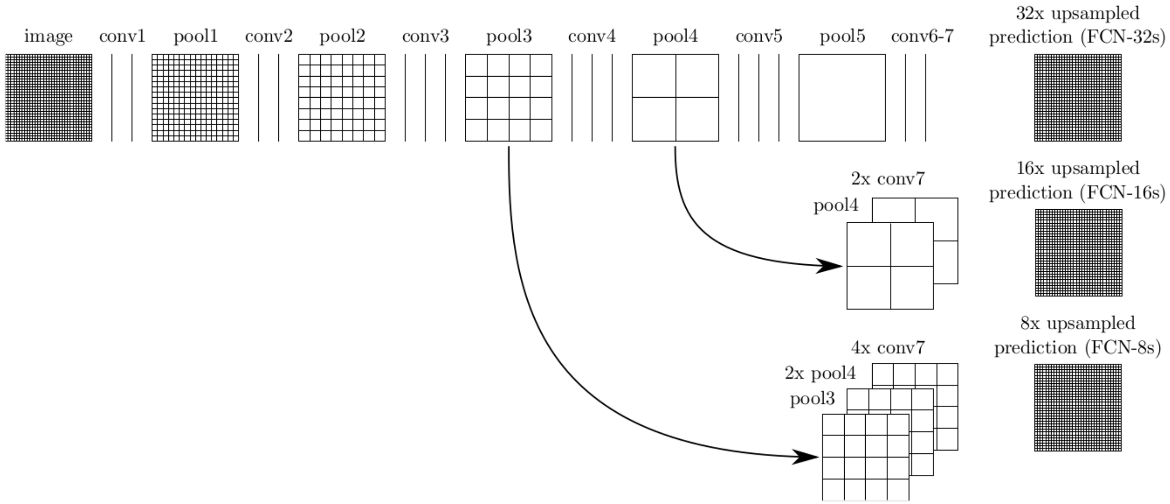
Source: SUMMER_story (n.d.)

FCN–8s

- *Long et al. (2015)*
- Converted classical classification networks to FCNs
- Originally designed for natural imagery

- Why selected
  - Successfully used by participants in ISPRS Semantic Labelling Challenge
  - Relatively simple to understand and to train
  - Focuses on capturing detail

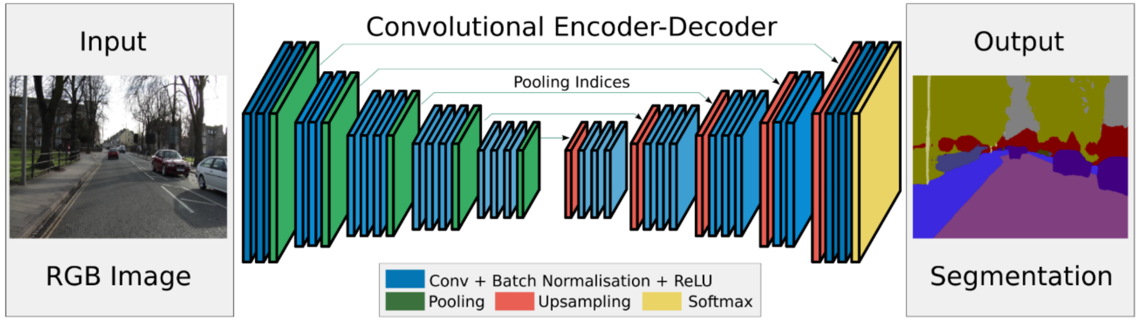
- Architecture
  - Replaced fully connected layers by **convolutional layers**
  - Learns deconvolution filters to perform upsampling



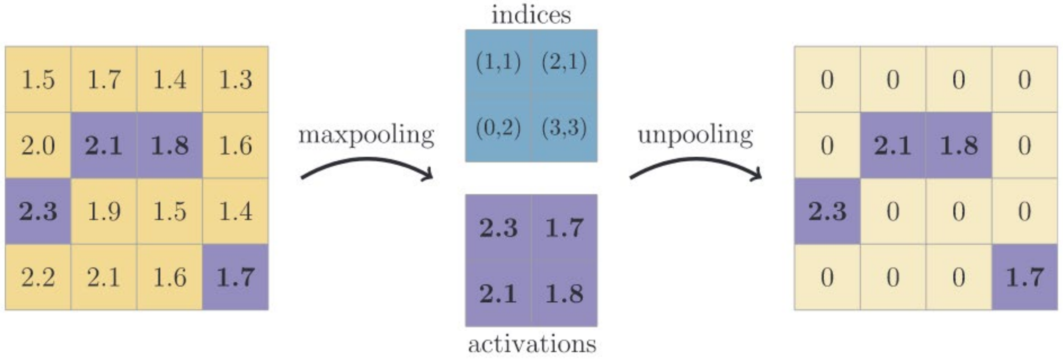
FCN-8s architecture (bottom) (Long et al., 2015)

SegNet

- *Badrinarayanan et al. (2017)*
- Originally designed for road scenery understanding (natural imagery)
- <u>Why selected</u>
  - Focused on improving boundaries
  - Similar semantic segmentation task
- <u>Architecture</u>
  - For every encoder layer: a corresponding decoder layer
  - Encoders pass on max-pooling indices which are used for upsampling



SegNet architecture (Badrinarayanan et al., 2017)



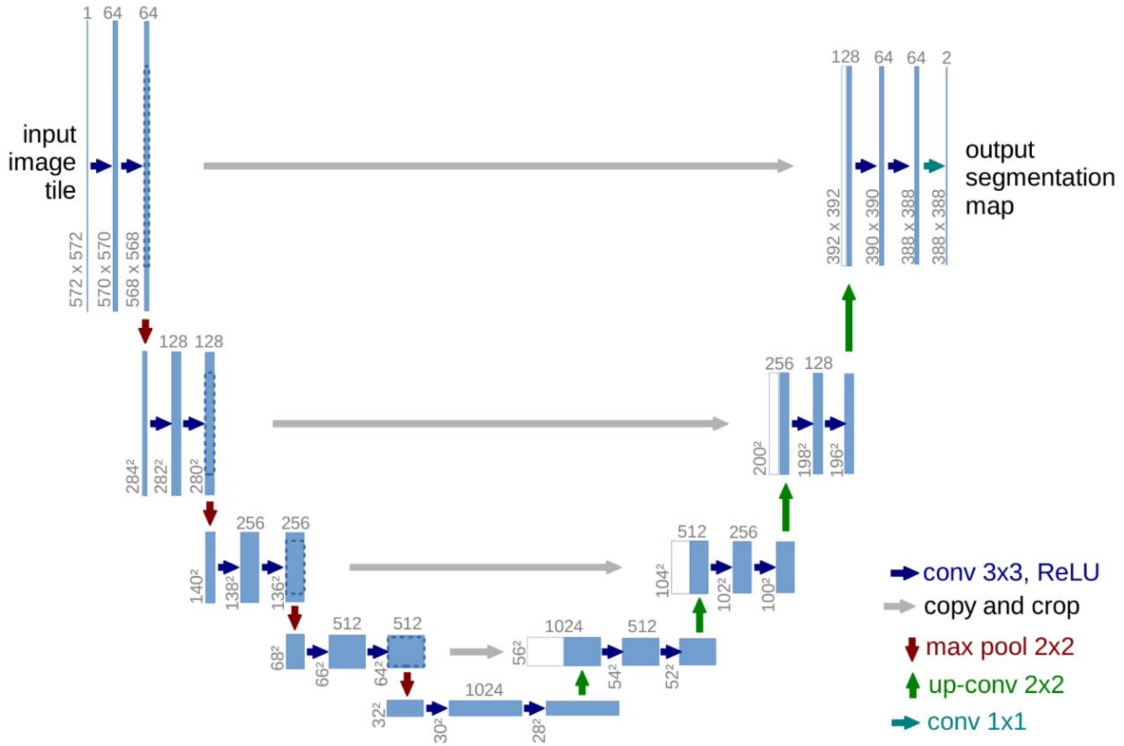
Max-pooling and unpooling on 4x4 feature map (Badrinarayanan et al., 2017)

U-Net

- *Ronneberger et al. (2015)*
- Originally designed for biomedical segmentation tasks
- Goal: work with very little training data

- Why selected
  - Often selected by high performing participants in Dstl Satellite Imagery Feature Detection Competition

- Architecture
  - Input differs from output dimensions
  - Transfers entire feature maps of encoder to matching decoders & concatenates them to the by deconvolution upsampled feature maps of decoder



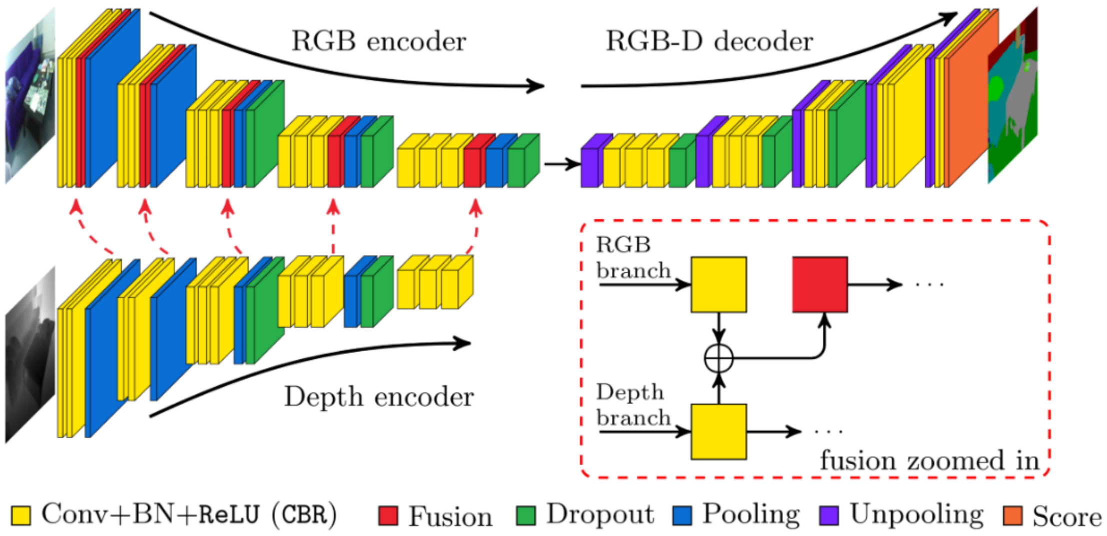
U-Net architecture (Ronneberger et al., 2015)

FuseNet-SF5

- *Hazirbas et al. (2017)*
- Originally designed for semantic segmentation of indoor scenes using RGB-D data
- **Fusion** of the depth information into RGB information instead of stacking
- Allows to learn depth (height) specific features

- Why selected?
  - Showed to outperform stacking approaches for indoor scenes with depth information
  - Successfully used on aerial imagery + LiDAR data (Audebert et al., 2018)

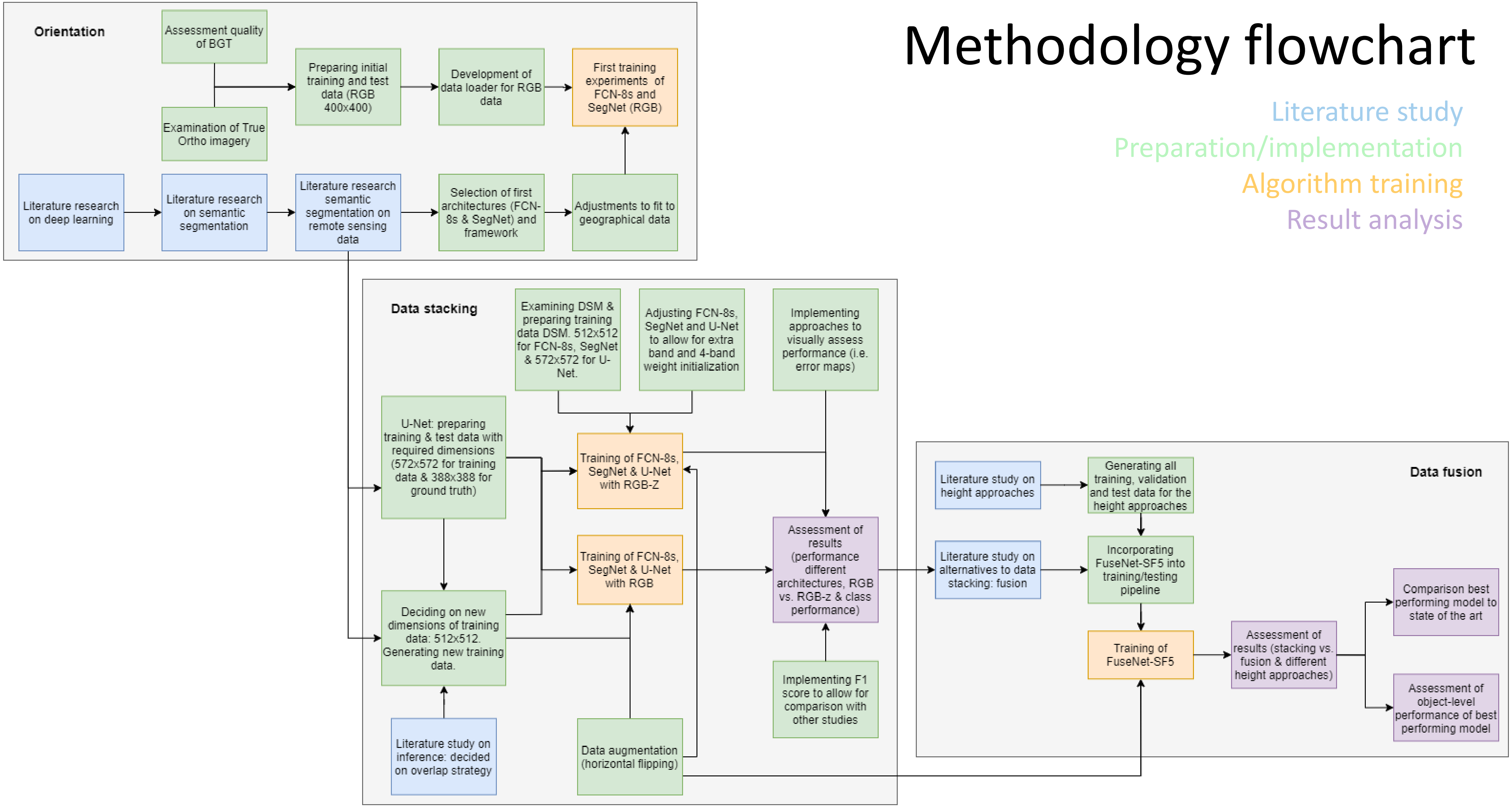
- Architecture
  - Two encoders: one for RGB & one for depth (or height)
  - Depth features are fused into RGB feature maps



Architecture of FuseNet-SF5 (Hazirbas et al., 2017)

Methodology flowchart

Literature study
Preparation/implementation
Algorithm training
Result analysis



Assessment BGT

+

+ Many different classes

+ Size and extent of dataset is large

+ Generally detailed geometry

+ Quality requirements are set

-

- Occasional boundary issues
- Different resolution
- “Begroeid” & “onbegroeid” mixed up

Conclusion: **quality** and **quantity** sufficient to serve as mask layer for ‘building’, ‘road’, ‘water’ and ‘other’

Deviating boundary

“Onbegroeid terreindeel” contains grass and trees

“Begroeid terreindeel” contains tarmac

Addition of extra band

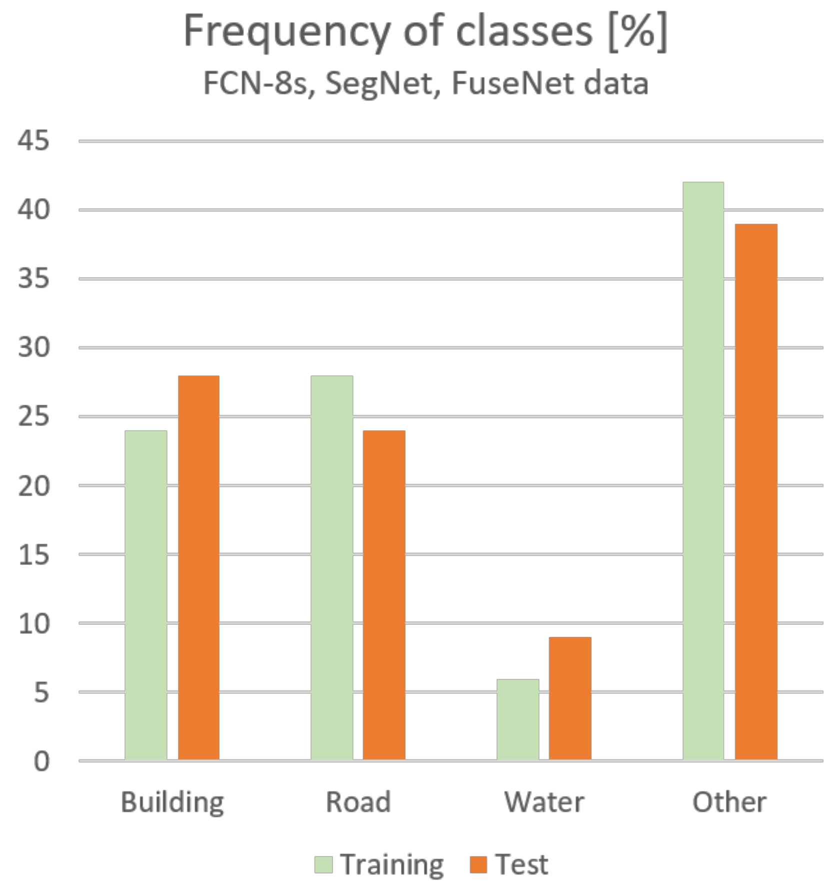
- How?
  - ➢ Change number of input channels!

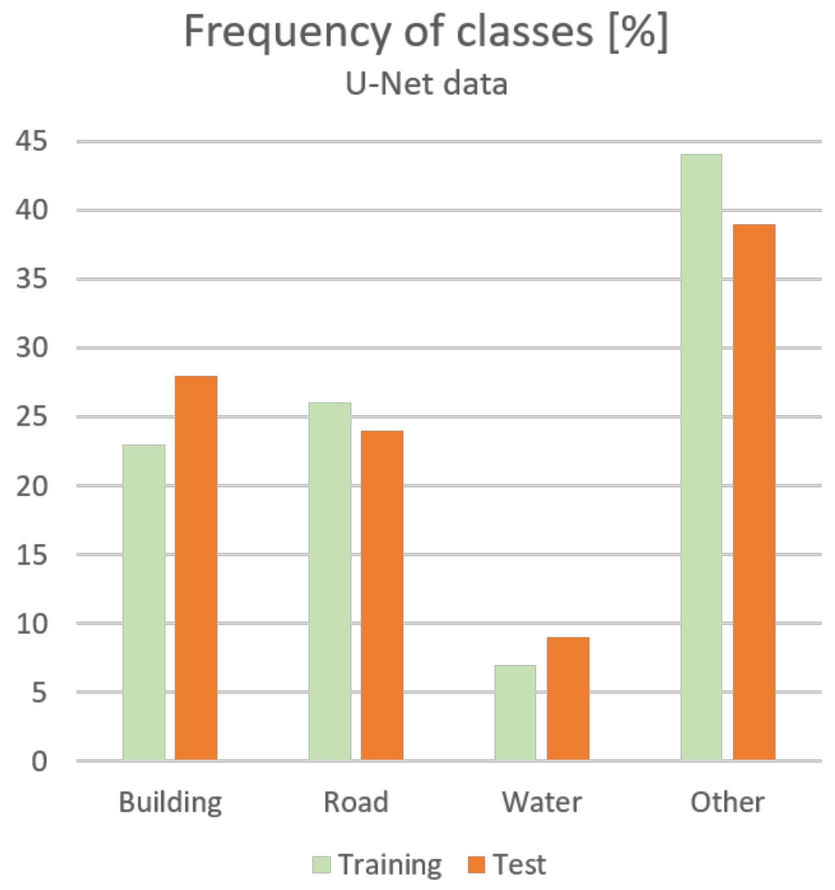
```python
self.conv_block1 = nn.Sequential(
    nn.Conv2d(4, 64, 3, padding=100),
    nn.ReLU(inplace=True),
    nn.Conv2d(64, 64, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(2, stride=2, ceil_mode=True),
)

self.conv_block2 = nn.Sequential(
    nn.Conv2d(64, 128, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.Conv2d(128, 128, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(2, stride=2, ceil_mode=True),
)

self.conv_block3 = nn.Sequential(
    nn.Conv2d(128, 256, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.Conv2d(256, 256, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.Conv2d(256, 256, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(2, stride=2, ceil_mode=True),
)

self.conv_block4 = nn.Sequential(
    nn.Conv2d(256, 512, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.Conv2d(512, 512, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.Conv2d(512, 512, 3, padding=1),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(2, stride=2, ceil_mode=True),
)

self.conv_block5 = nn.Sequential(
```

Pretrained weights

RGB	RGB-Z
FCN-8s, SegNet and **FuseNet**-SF5: VGG16 **U-Net:** Not available	**FCN-8s** and **SegNet**: VGG16 + random **FuseNet-SF5:** VGG16 + average VGG16 **U-Net:** Not available

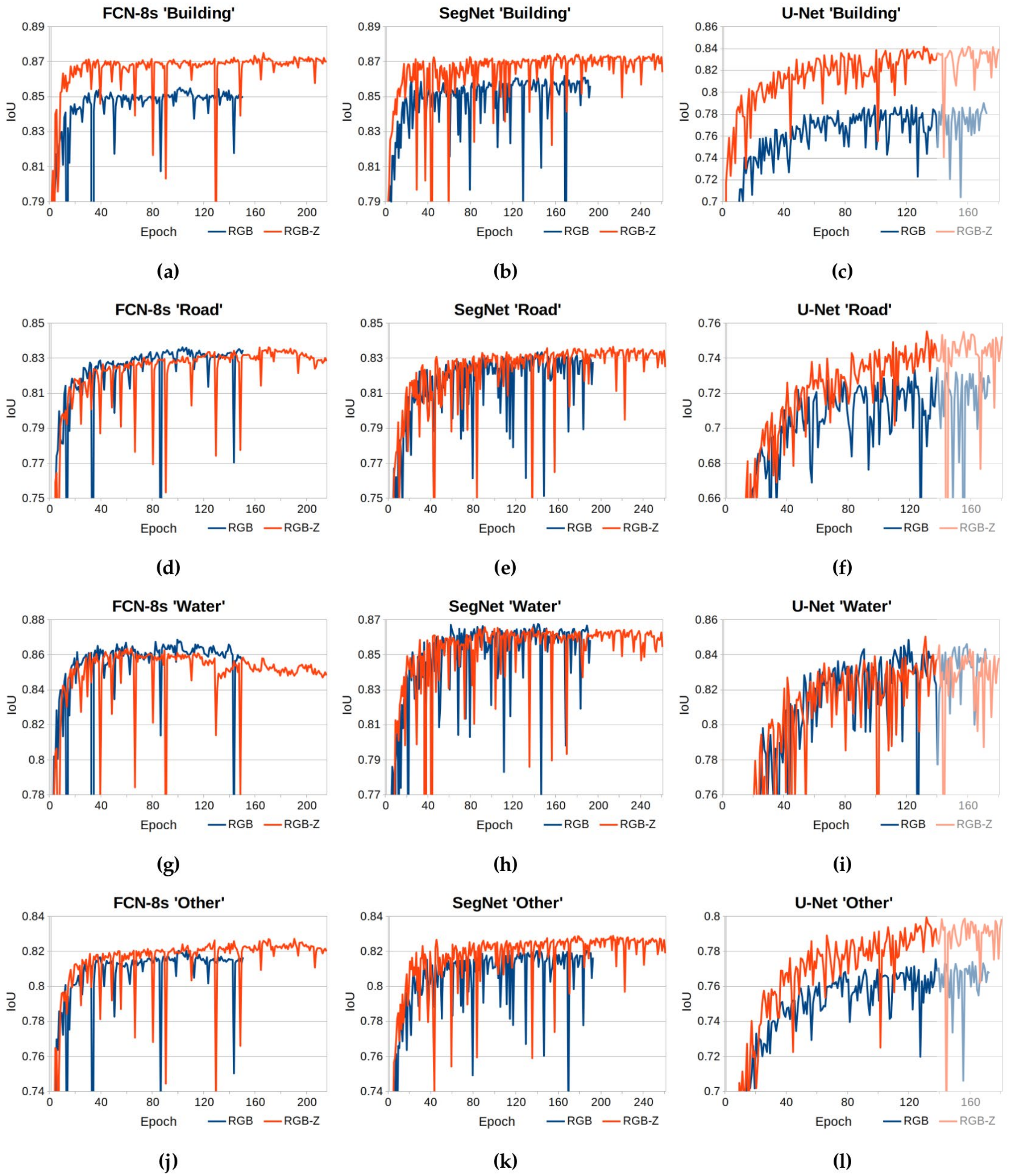
Class frequencies





The performance on the validation data,

achieved per class during training



Confusion matrices (1/2)

		Prediction			
		Building	**Road**	**Water**	**Other**
Actual	**Building**	**90.55**	1.04	0.09	8.32
	Road	1.19	**89.49**	0.14	9.19
	Water	1.98	0.70	**92.31**	5.01
	Other	4.15	8.01	0.67	**87.16**

SegNet (RGB)

		Prediction			
		Building	**Road**	**Water**	**Other**
Actual	**Building**	**91.77**	0.84	0.15	7.24
	Road	1.19	**89.51**	0.18	9.11
	Water	3.13	0.57	**91.58**	4.71
	Other	3.91	8.06	0.59	**87.44**

SegNet (RGB-Z)

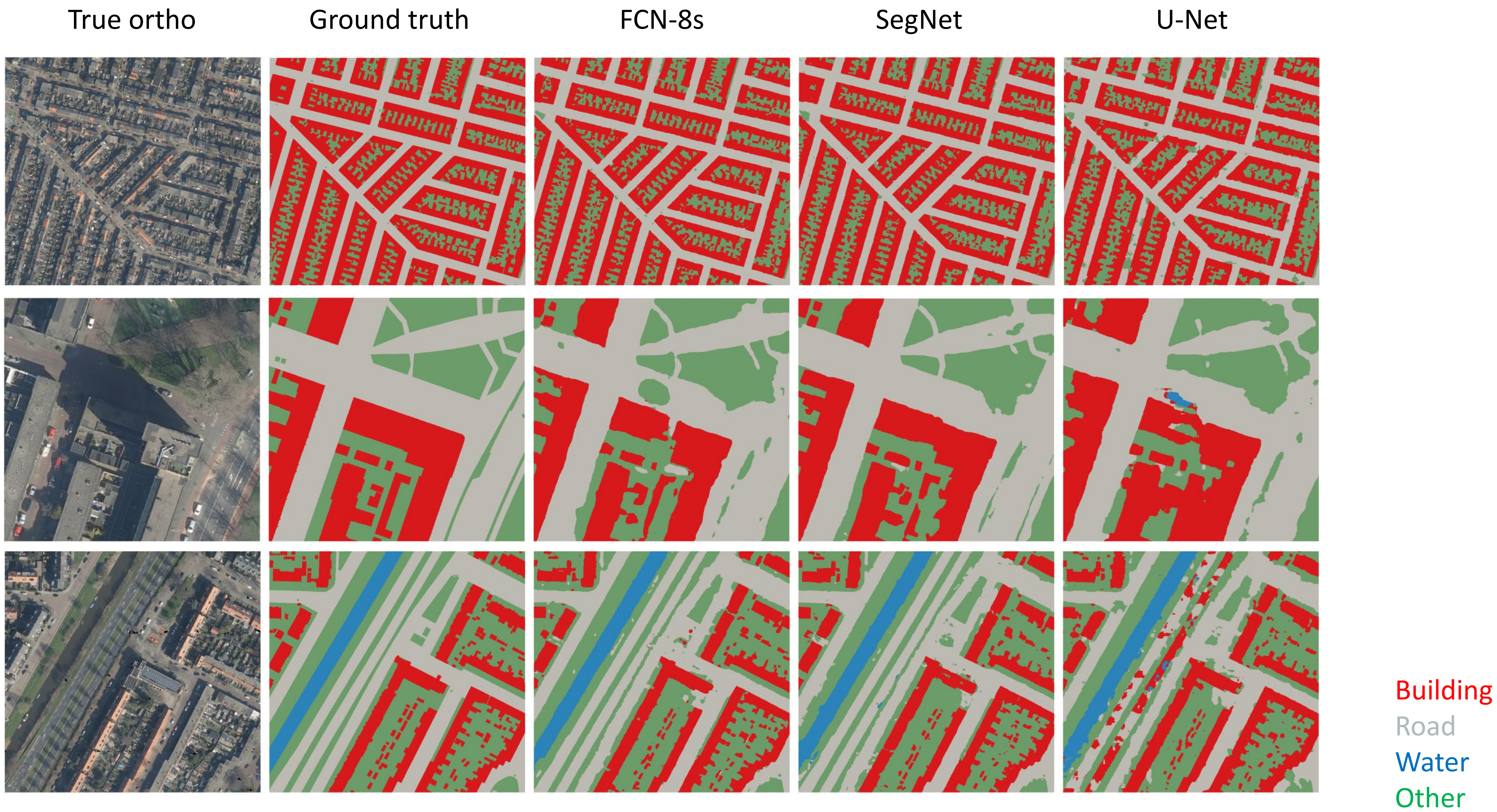
Confusion matrices (2/2)

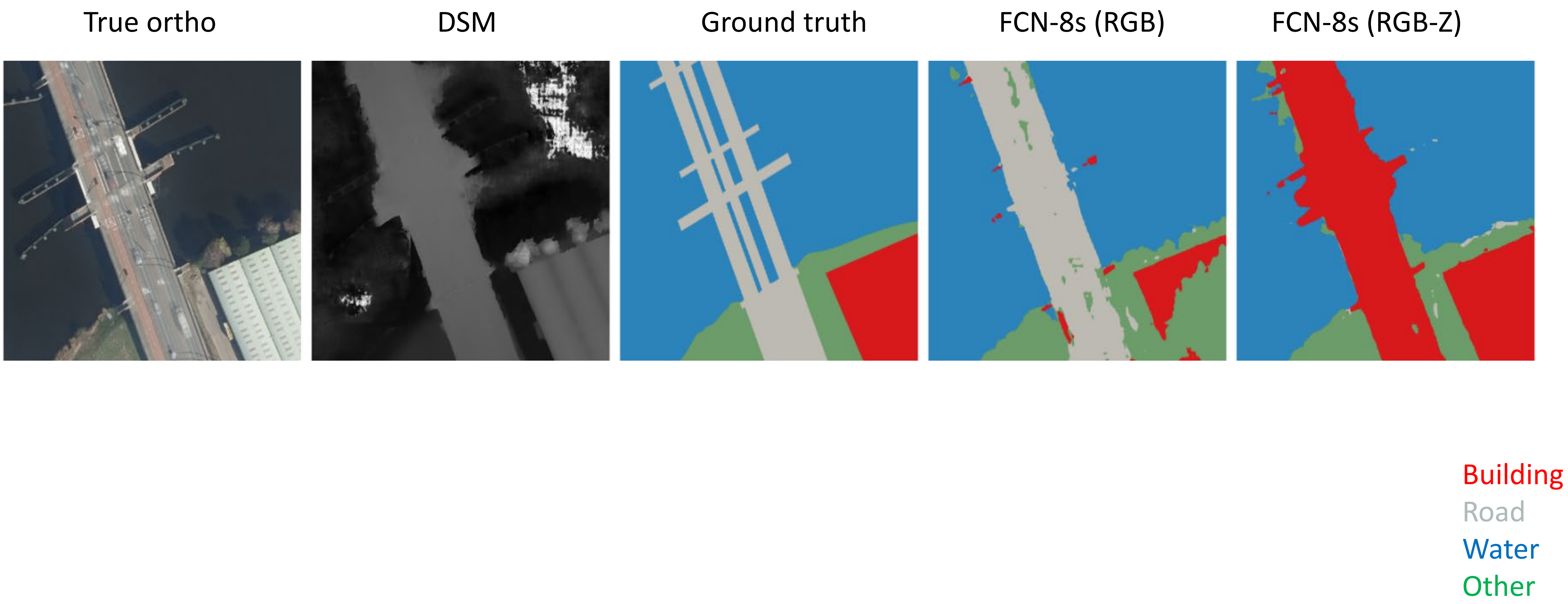
		Prediction			
		Building	**Road**	**Water**	**Other**
Actual	**Building**	**93.10**	0.92	0.04	5.94
	Road	1.18	**88.94**	0.07	9.81
	Water	1.34	0.82	**93.61**	4.23
	Other	3.78	8.02	0.47	**87.73**

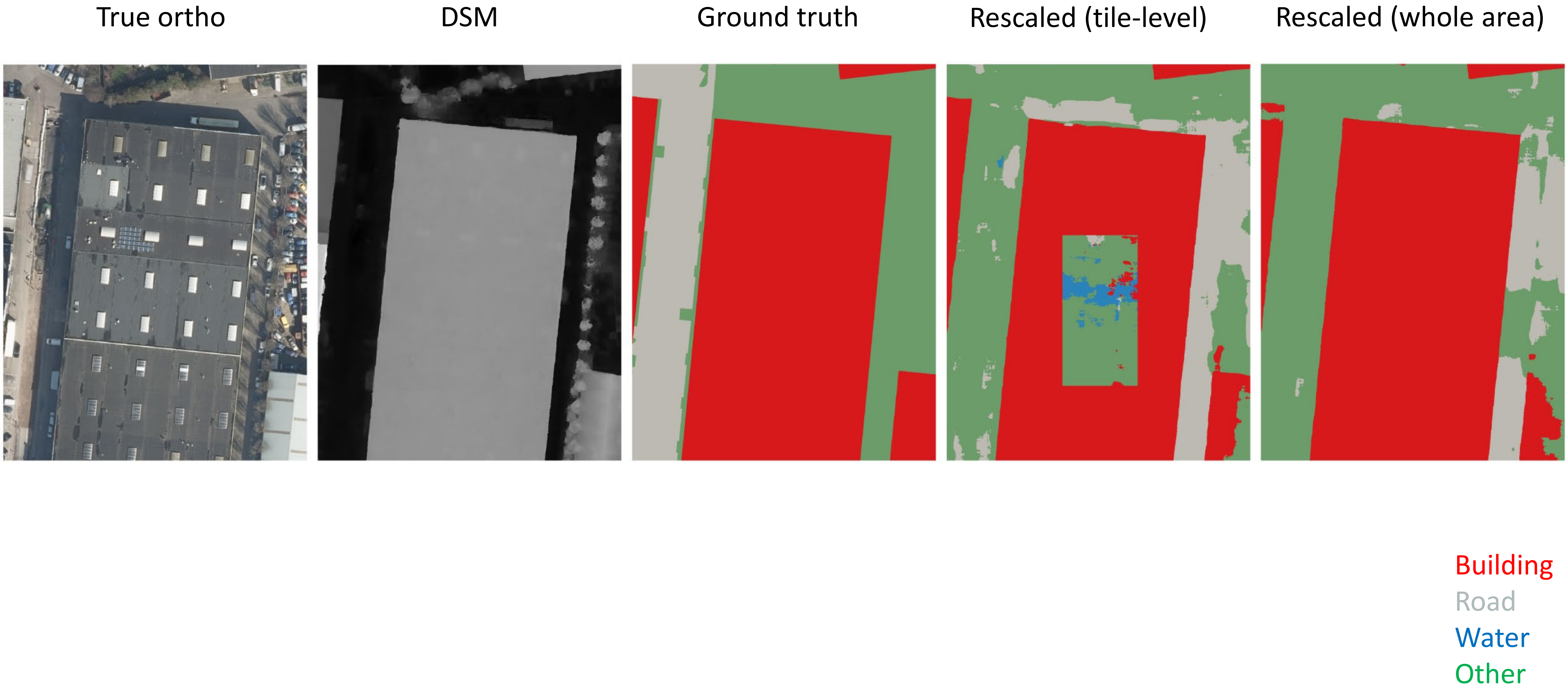
FuseNet-SF5 (absolute height)

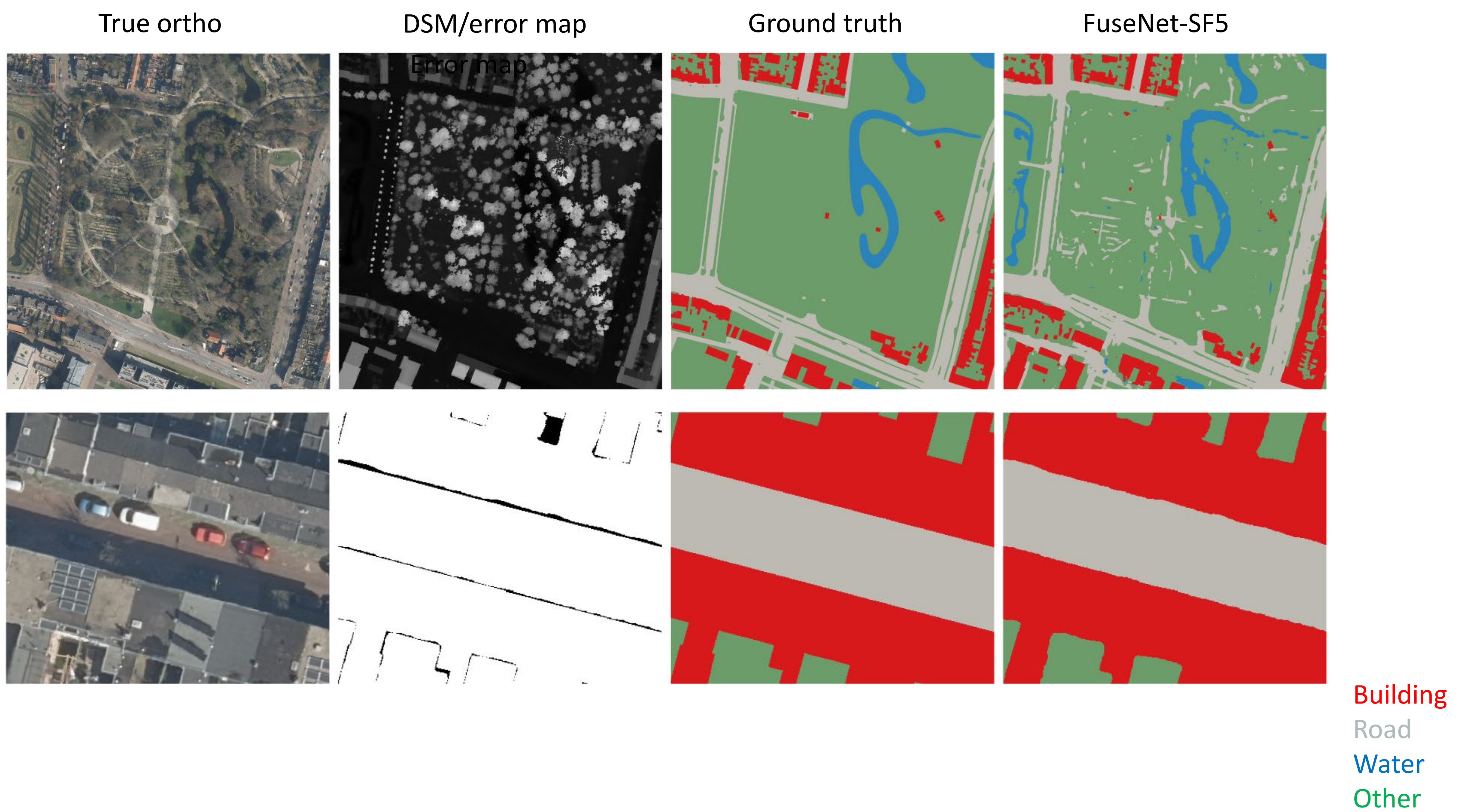
		Prediction			
		Building	**Road**	**Water**	**Other**
Actual	**Building**	**93.31**	0.74	0.05	5.90
	Road	1.47	**89.69**	0.27	8.57
	Water	1.31	0.44	**94.55**	3.69
	Other	3.59	7.95	0.60	**87.86**

FuseNet-SF5 (pixel-level, relative height)

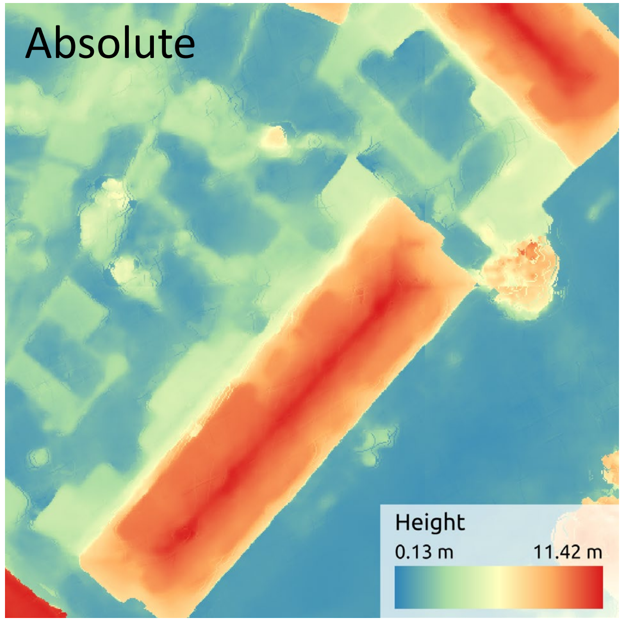

True ortho
Ground truth
FCN-8s
SegNet
U-Net
Building
Road
Water
Other

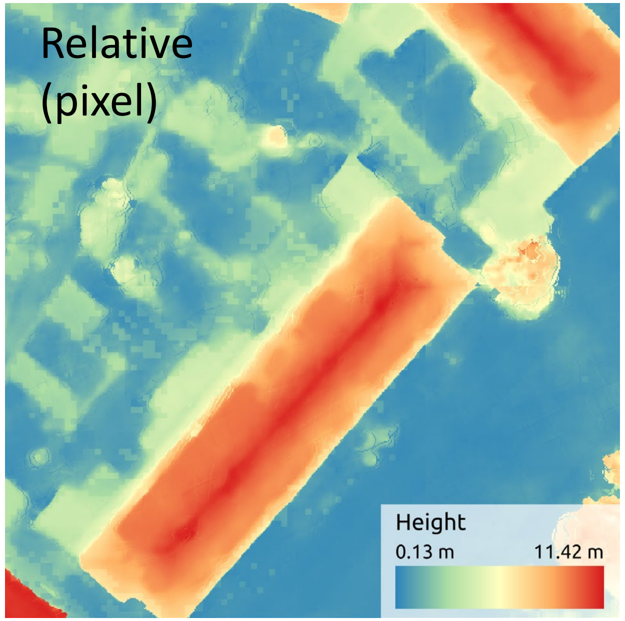

True ortho
DSM
Ground truth
FCN-8s (RGB)
FCN-8s (RGB-Z)
Building
Road
Water
Other

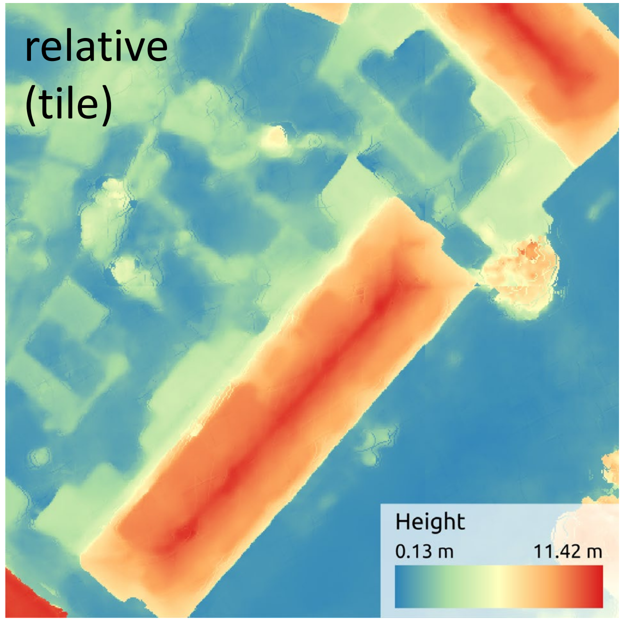

True ortho
DSM
Ground truth
Rescaled (tile-level)
Rescaled (whole area)
Building
Road
Water
Other

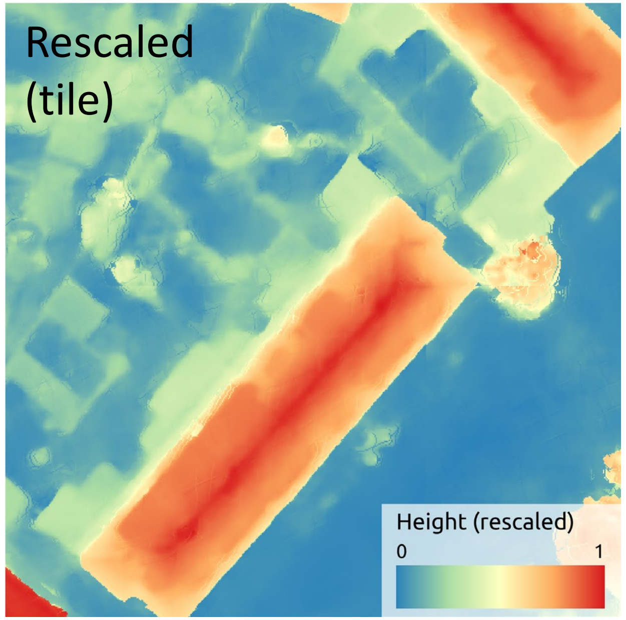

True ortho
DSM/error map
Error map
Ground truth
FuseNet-SF5
Building
Road
Water
Other

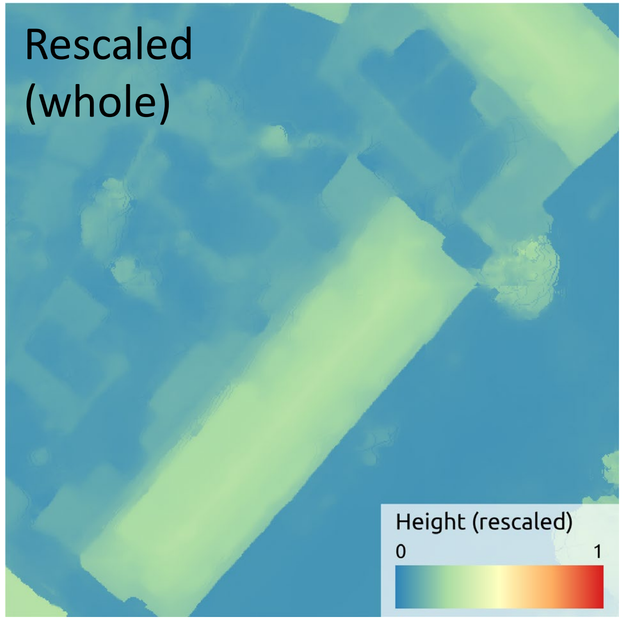
Height approaches



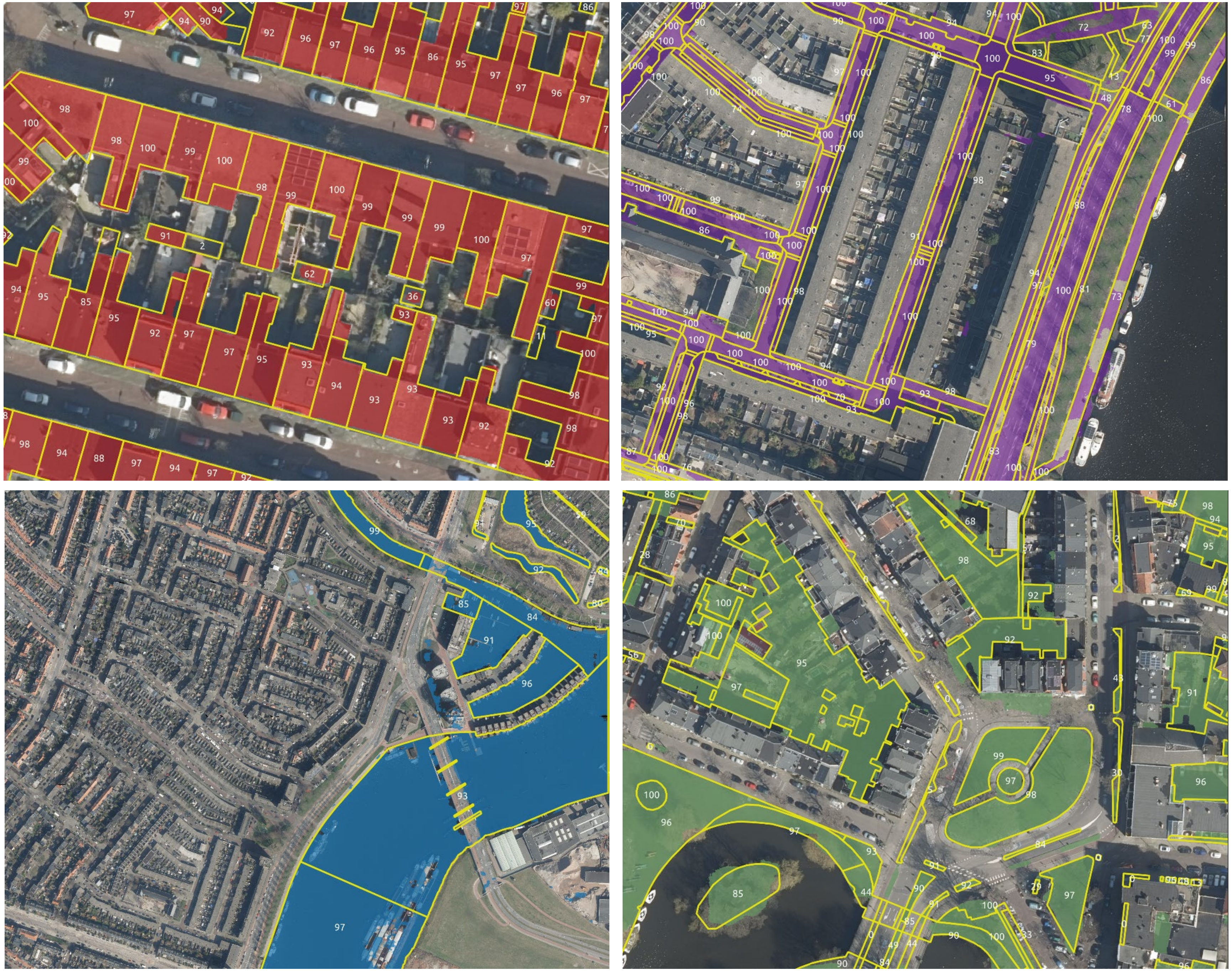








Object-level detection

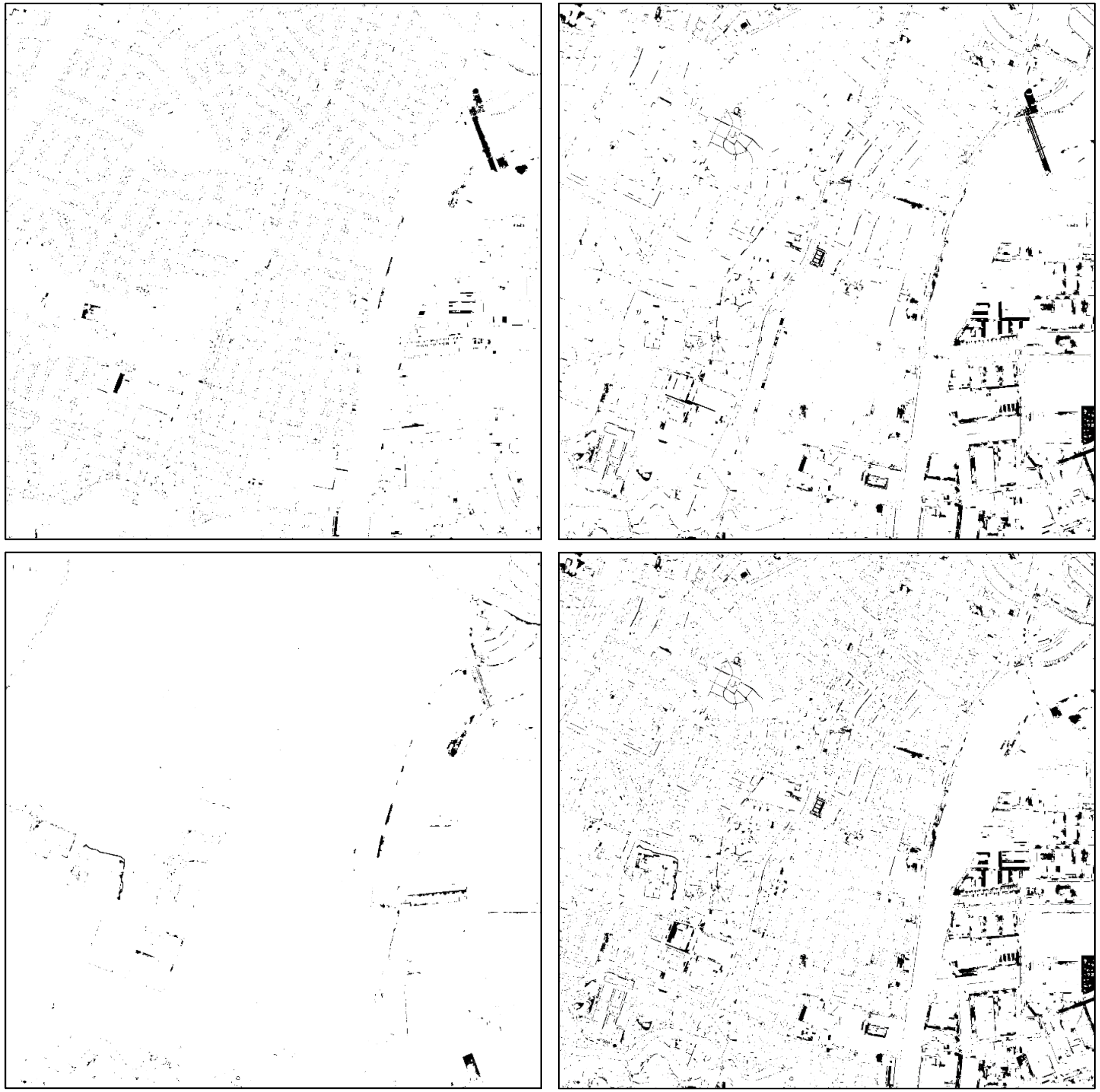


Comparison to related work

Method	**F1 Building**	**Note**
PB + FCN [Kampffmeyer et al., 2016]	**0.9586**	On validation data, with eroded ground truth boundaries.
HSN + OI erGT [Liu et al., 2017]	0.9466	On validation data, with eroded ground truth boundaries.
HSN + OI GT [Liu et al., 2017]	0.9237	On validation data, no eroded ground truth boundaries.
SegNet-RC [Audebert et al., 2018]	0.9450	On validation data, unclear if boundaries are eroded.
This study		
FuseNet-SF5-RHT (validation)	0.9436	On validation data, no eroded ground truth boundaries.
FuseNet-SF5-RHP (validation)	0.9429	On validation data, no eroded ground truth boundaries.
FuseNet-SF5-RHT (test)	0.9330	On test data, no eroded ground truth boundaries.
FuseNet-SF5-RHP (test)	0.9288	On test data, no eroded ground truth boundaries.

Results gained by related studies and this study for the class building.
PB = Patch based, **HSN** = Houreglass-shaped network, **OI** = Overlap inference,
GT = Ground truth, **erGT** = Eroded ground truth, **RC** = Residual correction,
RHP = Relative height (pixel-level), **RHT** = Relative height (tile-level).

Eroded error maps per class



Building

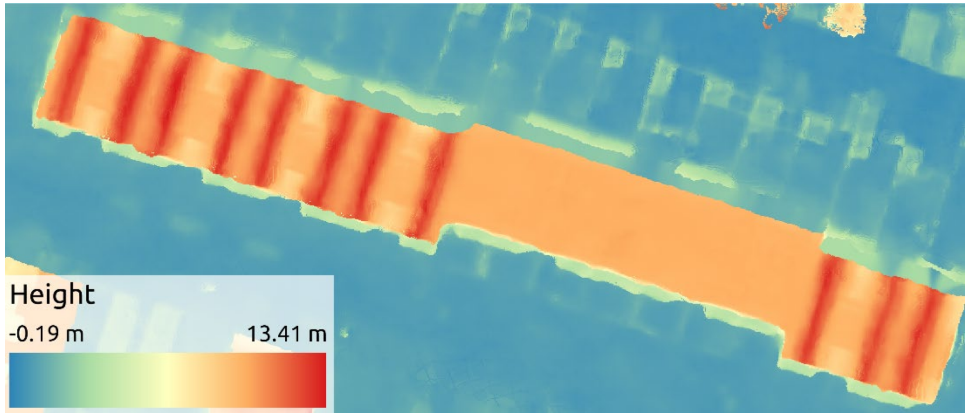
Road

Water

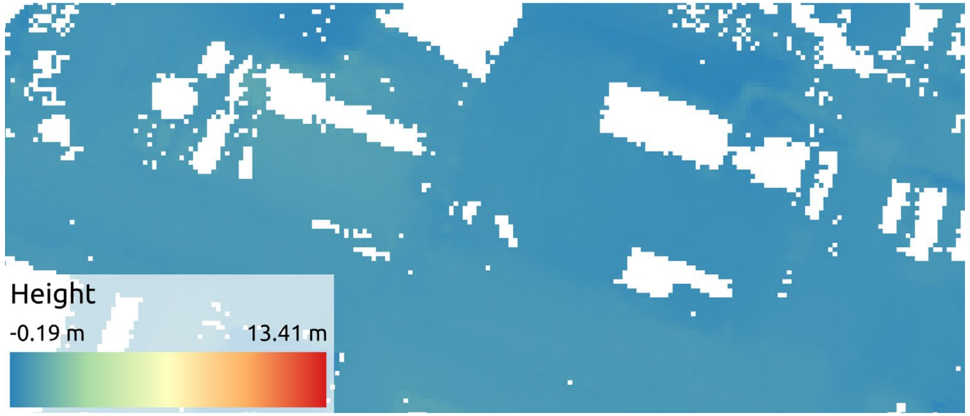
Other

Influence of interpolated holes

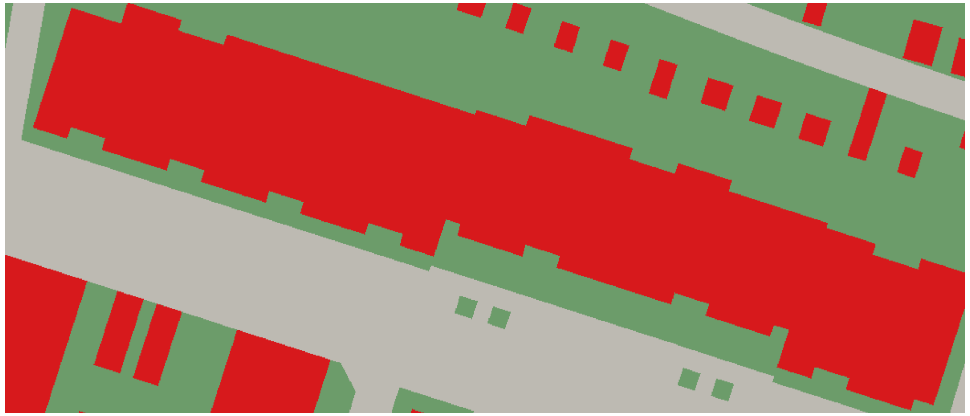
True ortho



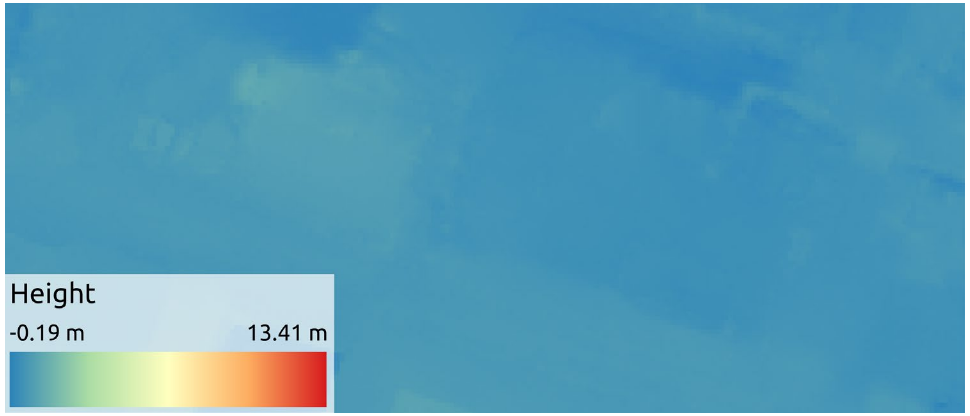
DSM



DTM



Ground truth



Interpolated DTM

FuseNet-SF5 using pixel-level, relative height

Recommendations supervisors P4

- Include a performance assessment based on the number of in/correctly classified objects, besides the current pixel-based measure.
- Improve on the DTM that you use for computing the relative heights. Instead of using DTM of AHN, generate DTM from used DSM.
- If you don't improve on the DTM, then it would be good if you could give a better assessment on to what extent the results for the pixel-based, relative height method are affected by the building contours in the DTM.
- Elaborate on how is the DSM created from READAR’s DM point cloud. Which algorithms are used for creating the DSM?