Assignment 2

Convolutional Neural Networks for Semantic Segmentation

Exercise for the course *Deep Learning in Computer Vision and Remote Sensing*

Summer term 2025

PD Dr.-Ing. Martin Weinmann (martin.weinmann@kit.edu)
M.Sc. Dennis Haitz (dennis.haitz@kit.edu)

1 Data and Training (14 P)

Semantic Segmentation based on Convolutional Neural Networks (CNNs) describes the task of assigning a class label to each pixel of an input image, using a CNN. Therefore, Semantic Segmentation can be thought of a per-pixel classification problem. In contrast to that, Assignment 1 dealt with the task of assigning a class label to the entire image.
The first task consists of training two CNNs of the same architecture with different channel configurations for each CNN. Regarding the CNN architecture, U-nets with a ResNet18 backbone are provided and pretrained to some extent on the dataset used in this task, with a patch size of 256×256 px. The pretrained models are provided as .pt files. For the channel configurations, one model needs to be trained with three channel input images (Figure 1 left) and the other with the three channel images and a stacked one channel image (Figure 1, left and middle), which results in a four channel input. Simliar to Assignment 1, you need to:

- write a dataset and dataloader class
- cast the input datatype to float32
- scale the value range to [0.0, 1.0]
- normalize the multispectral images (not the DSM!) with mean and std.

Additionally, you will likely not be able to train on the images as a whole because of memory constraints. Therefore you need to prepare your dataset so that you extract **neighbouring** image patches of size $n \times n$ px. Good choices are 256 or 512 for n. Go below for n if training goes out of memory. Keep in mind that this patches need also be extracted from the label image. The provided file *data-config.json* contains the information which of the images are used for training and how the colors are mapped to the class labels and names. Class names are not necessarily needed in this task. Train the models for 15-20 epochs, depending on the convergence behaviour (i.e. loss dropped to some extent). You can use a similar training configuration as in Assignment 1:

- Adam-Optimizer
- Learning-rate: 0.0001
- Batch size: 32
- Cross Entropy loss function.

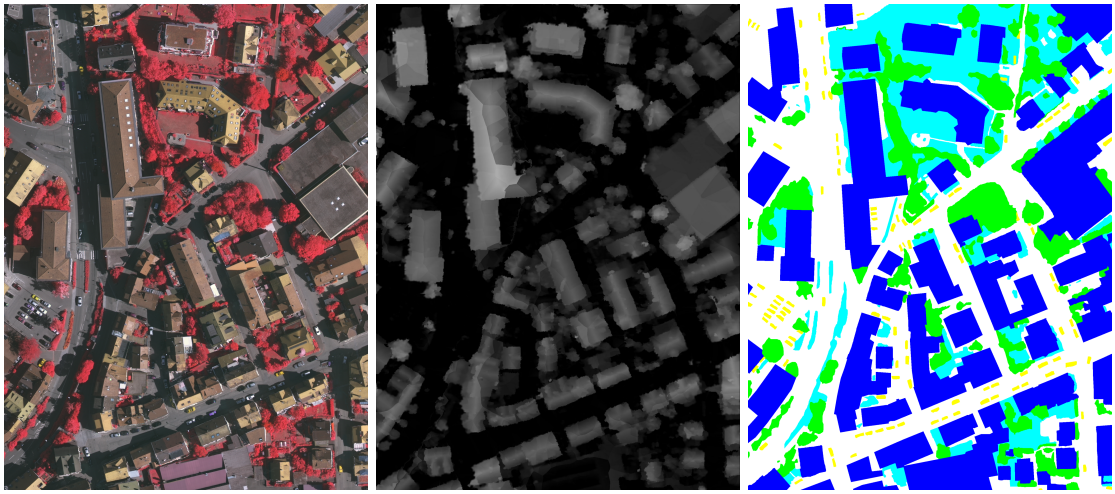

Figure 1: Samples of the dataset. Left: False color three channel image. Middle: Normalized DSM one channel image. Right: Label image.

Figure 2: Illustration of patches, cut out of the image, which then should be used as training input.

2 Testing (6 P)

Similar to Assignment 1, the classification metrics need to be calculated. Use both trained models from Task 1 and perform predictions on the test data. From the predicted outputs, calculate the Precision, Recall, F_1-Score, Overall Accuracy and the Confusion Matrix. To be precise, do this for every test image. Plot an example of a prediction from both models, using the same color scheme as the label images.

3 Questions (10 P)

Answer the following questions and send them as a PDF with your code.

1. What is a DSM as shown in Figure 1 in the middle? Give a short and precise definition and name methods and sensors for acquiring this kind of data. (2 P)

2. Adam and SGD (Stochastic Gradient Descent) are two widely used optimizers. Which core optimization technique do both have in common? (1 P)

3. A Huber-loss is a piece-wise loss-function consisting of L_1- and L_2-loss-functions, which accounts for robustness and smoothness. In which situations can a L_1-loss be favourable to the Huber-loss? (2 P)

4. Why does the four channel model perform better in Task 2? (1 P)

5. Name two other segmentation architectures besides U-net. Describe their essential methodology in 3-4 sentences each. (1 P)

6. Can Fully-Connected-Layers be substituted by Convolutional-Layers? (2 P)

7. Is it a good approach to also include test data into your training? Justify your answer from a Machine Learning standpoint. (1 P)