MULTISPECTRAL REMOTE SENSING

Master's Degree in Geospatial Engineering

LABORATORY PROJECT 2

DEEP LEARNING FOR BUILDING DETECTION USING HIGH-RESOLUTION IMAGES	
OBJECTIVE	Development of a complete workflow for building segmentation in Greater Lisbon using U-Net (PyTorch) and high-resolution orthophotos
DURATION	5 May 2026 – 26 May 2026
EVALUATION	Oral presentation – 2 June 2026
BIBLIOGRAPHY	The U-Net : A Complete Guide | Medium Building a U-Net Architecture for Image Segmentation with Python and Keras | by Musti Öztemiz | Medium Cook your First U-Net in PyTorch. A magic recipe to empower your image… | by Mostafa Wael | TDS Archive | Medium UNet for Building Segmentation (PyTorch) https://www.youtube.com/watch?v=IHq1t7NxS8k https://campus.datacamp.com/courses/deep-learning-for-images-with-pytorch/image-segmentation?ex=9

Building detection with U-Net workflow

STEP 1 **Area of interest (AOI) definition and data preparation in QGIS**	**1.1 Define the AOI** Select a sub-region inside the high-resolution orthophoto, mainly covered by buildings **1.2 Download building data from OSM**[1] **1.3 Load the high-resolution orthophoto** **1.4 Check CRS consistency** Both layers must use the same CRS; if different, reproject both to EPSG: 32629 **1.5 Clip data using AOI** Clip both the raster and vector data using AOI ***Output: AOI ortophoto and AOI building shapefile***
STEP 2 **Data preprocessing in PYTHON**	**2.1 Load input data (orthophoto and building shapefile)** **2.2 CRS check** **2.2 Convert the buildings vector file into a raster mask** **2.4 Save ground truth mask** ***Output: binary building mask (0= background, 1= building)***
STEP 3 **Tile generation for U-NET in PYTHON**	**3.1. Generate training patches** Split orthophoto and mask into tiles (256 x 256 pixels) ***Output: image tiles (X) and mask tiles (y)***
STEP 4 **U-Net model training in PYTHON**	**4.1 Define the model architecture** Encoder: ResNet34 (pretrained on ImageNet) Decoder: U-Net with skip connections **4.2 Training configuration** Loss function: BCE + Dice Optimizer: Adam (lr= 1e-3) Epochs: 10 to 30 ***Output: trained U-Net model for building segmentation***

[1] https://download.geofabrik.de/europe/portugal.html

STEP 5 **Prediction and post-processing in PYTHON**	***5.1 Sliding window prediction*** *Split the image into overlapping patches (e.g., 256 x 256); run prediction for each patch; store outputs in a full-size probability map* ***Output: raw probability map (continuous values 0-1)*** ***5.2 Seamless reconstruction*** *Combine overlapping predictions by averaging overlapping regions for a smooth output (removes tile boundary artifacts)* ***Output: seamless probability raster (no tile borders)*** ***5.3 Thresholding (binary building map)*** *Convert probabilities to classes by applying a threshold (commonly 0.5); if >= 0.5, building (1), else background (0)* ***Output: binary building map*** ***5.4 Export the prediction as GeoTIFF*** ***Output: Geotiff file fully compatible with QGIS and GIS workflows*** ***5.5 Evaluate the model performance*** *Compare the predicted map versus the ground truth mask* • *IoU (spatial overlap quality)* • *Dice (segmentation similarity, especially effective for buildings)* • *Accuracy (pixel-wise correctness)* ***Output: quantitative performance report***