Development of a Multi-Band Convolutional Neural Network with an Applied Example for Individual Tree Crown Delineation

Masterarbeit

vorgelegt von
You-De Chen
Immatrikulationsnummer 3604131
Studiengang M. Sc. Physische Geographie

Arbeitsgruppe Umweltinformatik
Fachbereich Geographie
Philipps-Universität Marburg

eingereicht am
07. Februar 2025

betreut und begutachtet von
Dirk Zeuss und Lisa Bald

Development of a Multi-Band Convolutional Neural Network with an Applied Example for Individual Tree Crown Delineation

Master's thesis by You-De Chen (you-de.chen@outlook.com) on 07. February 2025

Abstract—**Convolutional neural networks are popularly used in image analysis. However, many existing computer vision models need adaptation to be applied on Earth observation data, which usually contain more than three bands and are collected from multiple sources. Data fusion has the potential to address this challenge. In the context of individual tree crown delineation, it may also fill the research gap of delineating overlapping tree crowns, contributing to better tree management and research especially in tropical areas. I developed a workflow to train U-Net models with any number of bands and applied it to a benchmark dataset. During the training, testing, and delineation phases, I conducted a series of experiments to enhance the performance, including band selection, the use of a weighted loss function, U-Net model depth, tile overlap distance, and delineation tuning parameters. The final eight-band model reached a validation accuracy of 0.8764, outperforming that of a model using the usual red, green, and blue bands by 0.15. The final delineation result achieved an overall precision of 0.3634 and recall of 0.4222, which were comparable to some traditional image processing methods. With few training data, the model predicted tree crown pixels relatively well, but the delineation of overlapping tree crowns remained a challenge. The source code is available at https://github.com/AlexCYD/multi-band-CNN-for-individual-tree-crown-delineation.git.**

1. Introduction

Convolutional Neural Network (CNN) is a specific type of Artificial Neural Network (ANN) and is often used in image analysis for pattern recognition [1]. The basic structure of ANNs comprises an input layer, an output layer, and one or more hidden layers [1]. A large number of parameters resulting from dense connections hinders the application of ANNs on larger computer vision tasks due to overfitting, long training time, and limited computational resources [1]. CNNs feature convolutional layers which are connected only to a small part of the preceding layer, reducing the total number of parameters and thus the effect of overfitting [1], [2]. CNNs are therefore more appropriate than traditional ANNs to analyze images having higher input dimensions (height, width, and number of bands) [1].

CNNs have been widely used in processing Earth observation images [3], [4]. However, CNNs originally developed for computer vision tasks can often not be directly applied to Earth observation data, as they often contain more than just Red, Green, and Blue (RGB) bands [5]. Furthermore, while computer vision data often originate from a single sensor and platform, Earth observation remote sensing data are collected from various sensors (e. g. RGB, multispectral, hyperspectral, Light Detection And Ranging (LiDAR), radar) and platforms (e. g. satellite, aircraft, drone, terrestrial station) [4], [5]. Extra model adjustment or data pre-processing may pose a barrier for CNN application in Earth observation.

To leverage data from multiple bands and sources, data fusion methods at different modeling levels have been developed [3], [4], [6]. At the input level, CNN architectures can be adjusted to accept more than three bands [6]; at the feature level, feature maps (output of CNN hidden layers) obtained from different input bands can be concatenated for use in later CNN layers [4]; and at the result level, the final prediction can be decided by, majority voting for example, using outputs from multiple input bands and CNN models [3].

In recent years, CNNs have been numerously applied in Individual Tree Crown Delineation (ITCD), whose purpose is to automatically identify tree crown extents from Earth observation images, saving effort of field work and manual visual assessment of images [6]. Results of ITCD provide valuable and up-to-date information such as tree species, health and growing status, canopy cover, and above-ground biomass for research fields including ecology and climate [7]. They also contribute to more accurate management of trees in various environment, such as natural forests, plantations, and urban areas [6].

CNN-based ITCD methods have shown higher accuracy in larger study areas compared to traditional image processing and traditional machine learning ITCD methods [7]. Various CNN model architectures were developed for different computer vision tasks [3]. ITCD can be classified as an instance segmentation task in computer vision and corresponding CNN model architectures can be used [3]. Architectures designed for semantic segmentation tasks may also be employed to ITCD, but post-processing to transform pixel-wise probability results into individual tree crown polygons is needed [7].

To address challenges such as overlapping tree crowns and shadows, particularly in dense forests, data fusion methods possess significant potential for advancing ITCD [6], [7]. However, only a limited number of (CNN) studies have leveraged data from multiple sources so far [6], [7]. Therefore, the objective of this study was to develop a workflow for training multi-band CNN models with any number of bands to investigate whether fusing data from multiple sources could achieve better performance in ITCD than using common RGB images from a single source. Furthermore, I conducted experiments at training, testing, and delineation phases in an attempt to improve the results.

2. Data

The ITCD benchmark dataset paper [8] released training and testing datasets. In this study, I used their training data version 0.2.2 [9] and testing data version 1.8.0 [10]. I used RGB images along with corresponding hyperspectral images, canopy height model (CHM) images, and tree crown annotations. LiDAR point cloud data were also available, but I did not use them directly in this study.

Both training and testing remote sensing data originated from the National Ecological Observatory Network (NEON) Airborne Observation Platform (AOP) of the United States. Data were collected by aircraft flying 1 km above the ground during leaf-on-conditions from May to October in 2018 and 2019 [8]. NEON's hyperspectral images included 426 bands ranging from 383 to 2512 nm with an interval of 5 nm [8], [10]. CHM images of NEON were derived from LiDAR data with a density of about 5 points per m^2 [8]. The authors of the benchmark dataset paper [8] cropped selected NEON images and deliberately kept RGB images that were slightly distorted due to georectification with LiDAR data to highlight the challenge of aligning data from different sensors [8]. They drew rectangular bounding boxes for every tree crown in the training dataset disregarding its health status by assessing RGB images [8]. In contrast, they annotated tree crowns in the testing dataset by comparing RGB, hyperspectral, and LiDAR data and reviewed the correctness of these annotations more thoroughly [8]. Annotations with a tree height lower than 3 m were removed [8], [11].

In the training and testing datasets, the number of files within CHM, RGB, hyperspectral, and annotation folders differed. To get complete file sets in both training and testing datasets, I identified common filenames across their RGB, hyperspectral, CHM, and annotation files. The filenames contained information about NEON site name (Appendix Table A.1), plot numbering, and year of data collection. There were 16 complete file sets for training and 189 for testing. After analyzing their metadata, I excluded SJER_062_2018 and TALL_043_2019 in the testing dataset because their CHM images were much smaller in extent than their RGB and hyperspectral counterparts. Furthermore, I omitted testing data SJER_016_2018 because it overlapped with training data. As a result, 16 complete file sets for training and 186 for testing were available for further processes (Appendix Table A.2).

Among the complete file sets, RGB images had a resolution of 0.1 m, while that of hyperspectral and CHM images was 1 m (Table 1). The size of training images ranged from 90 x 116 m to 1 x 1 km and testing images were all about 40 x 40 m. No coordinate reference system was assigned to testing hyperspectral images, but I assumed it was the same as their corresponding RGB images. Furthermore, I aligned the extent of CHM and hyperspectral images to their RGB images in both training and testing data because they did not always match exactly.

More than 20,000 and 6,000 tree crowns were annotated in the complete file sets of the training and testing data of this study respectively and the main land covers were evergreen and deciduous forest (Appendix Table A.2). Training and testing data originated from NEON sites distributed across the United States (Fig. 1).

TABLE 1
Metadata of Training and Testing Remote Sensing Images
RGB: Red, Green, and Blue. CHM: Canopy Height Model

Image type	Number of bands	Resolution
RGB	3	0.1 m
CHM	1	1 m
Hyperspectral	426	1 m

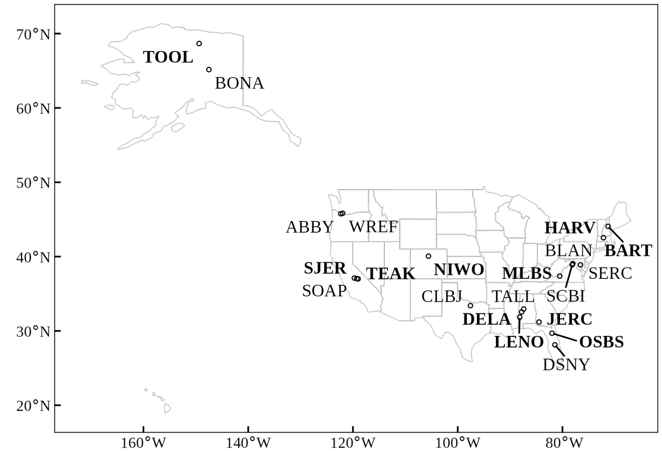

Fig. 1. Locations of all NEON sites in the training (bold) and testing data across the United States. More sites were located on the East Coast. Full site names are listed in Appendix Table A.1.

3. Methods

I conducted data analysis and processing with R version 4.3 [12]. Models were trained on Ubuntu 24.04 with two NVIDIA GeForce RTX 2070 GPUs of 8 GB memory using *TensorFlow for R* version 2.16 [13] and R *keras* version 2.15 [14], which were interfaces to Python *TensorFlow* version 2.15 and Python *Keras* version 2.15. I relied on the R *terra* package [15] for image processing and analysis, *lidR* [16], [17] for delineation, and *NeonTreeEvaluation* [18] for delineation assessment. The source code of this study is available at https://github.com/AlexCYD/multi-band-CNN-for-individual-tree-crown-delineation.git.

3.1 Data pre-processing

When training or predicting with CNNs, large images of various sizes should be split into small tiles of the same size due to limited memory [6] and the design of the model input. For training data, I first generated 128 x 128 pixels training target tiles at a resolution of 0.1 m from RGB images and tree crown annotation files. In these tiles, a value of 1 represented tree crown pixels and 0 represented non-crown pixels. For all images whose size was not an integer multiple of 128 pixels, I removed residual pixels evenly on all sides. To reduce the possible negative effect on learning with tiles containing only tree crown or only non-crown pixels, I used only all 9061 tiles with both types of pixels (Appendix Table A.2). This corresponded to about 25% of all training target tiles that could potentially be generated. For testing images, I expanded them to the next integer multiple of 128 pixels using mirroring [19] before splitting them into target tiles. This expansion ensured that output prediction results covered the whole extent of the input images.

To generate predictor tiles, I first cropped RGB, hyperspectral, and CHM images to their corresponding target tiles. Then, I replaced missing values by mean values of their neighboring eight cells repeatedly until no missing values were present in the images. Furthermore, I resampled hyperspectral and CHM images to the resolution of RGB images using bilinear interpolation.

I selected and combined 15 bands for each training predictor tile (Table 2). Apart from RGB and CHM bands, I added three hyperspectral bands with the numbering 11 (433 nm), 55 (653 nm), and 113 (944 nm). They were used in the manual annotation of tree crowns in the benchmark dataset and were able to distinguish nearby trees of different types [8], [20]. I also chose to include an NIR band (band 96, 858 nm) because green plants show high NIR reflectance [21]. It was used in several previous studies [21], [22], [23]. Moreover, I calculated seven vegetation indices applied in earlier studies including Normalized Difference Vegetation Index (NDVI), Enhanced Vegetation Index (EVI), Atmospherically Resistant Vegetation Index (ARVI), Physiological Reflectance Index (PRI), Normalized Difference Lignin Index (NDLI), Soil-Adjusted Vegetation Index (SAVI), and Green Normalized Difference Vegetation Index (GNDVI) [22], [23], [24].

These vegetation indices may be useful in highlighting specific characteristics of trees that are not apparent in RGB images and can thus separate neighboring trees apart. NDVI presents the greenness of trees [25]. EVI [25], ARVI [26], SAVI [27], and GNDVI [28] aimed to improve NDVI by reducing influences from the soil or the atmosphere. PRI highlights canopy photosynthetic activity [29] and NDLI shows canopy lignin content [30]. I chose hyperspectral bands 84 (798 nm) and 58 (668 nm) to calculate SAVI and selected bands 96 (858 nm) and 31 (533 nm) for GNDVI. For other vegetation indices, I used the hyperspectral bands specified in the NEON document for the calculation [25]. Finally, I normalized all bands to the range of 0 and 1 and saved them as TIF images. Appendix Fig. A.1 includes a visual example of all bands in a training predictor tile along with its corresponding target tile.

3.2 Model architecture

While CNN models for instance segmentation or object detection tasks can also be used [6], I chose to use U-Net, a semantic segmentation CNN, due to its potential in segmenting touching objects [19]. I used the R implementation from [31], which had a modular design, allowing easy adjustment of the depth of U-Net. The deeper a U-Net is, the more levels it has and the deeper the U-shape is in the model architecture (Appendix Fig. A.2). Defining a ”block” as two consecutive 3 x 3 padded convolution with Rectified Linear Unit (ReLU) activation, each level in the contracting path contained one block and a 2 x 2 max pooling with 2 x 2 stride. The number of feature maps started with 64 and doubled as the path went deeper. At the bottom of the U-shape was a dropout of 0.5 to reduce overfitting and a block. Each level in the expansive path contained a transpose layer of 2 x 2 padded convolution with 2 x 2 stride, concatenated with the corresponding feature maps from the contracting path, followed by a block. Finally, there was an 1 x 1 convolution with sigmoid output activation.

TABLE 2
BAND NAMES IN EVERY TRAINING PREDICTOR TILE

Abbreviation	Full name
R	Red
G	Green
B	Blue
CHM	Canopy Height Model
11	Hyperspectral band number 11
55	Hyperspectral band number 55
113	Hyperspectral band number 113
NIR	Near-Infrared
NDVI	Normalized Difference Vegetation Index
EVI	Enhanced Vegetation Index
ARVI	Atmospherically Resistant Vegetation Index
PRI	Physiological Reflectance Index
NDLI	Normalized Difference Lignin Index
SAVI	Soil-Adjusted Vegetation Index
GNDVI	Green Normalized Difference Vegetation Index

3.3 Training

Python *TensorFlow* image processing functions supported only few image formats with a certain number of bands [32]. For processing TIF images with any number of bands in this study, I needed to create function to extract pixel values for training CNN models. As for adjusting the architecture of the model for it to accept any number of bands, it was as simple as changing a single parameter in the U-Net model that controls the input dimensionality.

The large number of parameters in the training process can occupy much memory, resulting in long training time. By using a data type of lower precision for storing less important parameters while keeping decisive parameters stored in higher precision, the model is able to achieve the same quality at lower memory usage and shorter training time [33]. Although I used this mixed precision setting, I still could not train the model with all 9061 tiles due to the out of memory problem. Therefore, I decided to randomly selected 100 tiles (Appendix Table A.2) for the training process, where I used 80 tiles for training and 20 for validation.

Augmentation is a technique to avoid overfitting by introducing more variability to the images [34]. I used two augmentation methods popular in ITCD studies [6] sequentially by first randomly flipping training data horizontally and vertically, and then randomly rotating them between ± 45 degrees. Augmented data were added to the original data, doubling the number of training tiles to 160.

Following [19], I set batch size to one, used Stochastic Gradient Descent (SDG) optimizer with a momentum of 0.99, and initiated kernel weights with the HeNormal function [35]. I used a learning rate of 0.001 in the optimizer. The number of total epochs was large enough and training terminated early

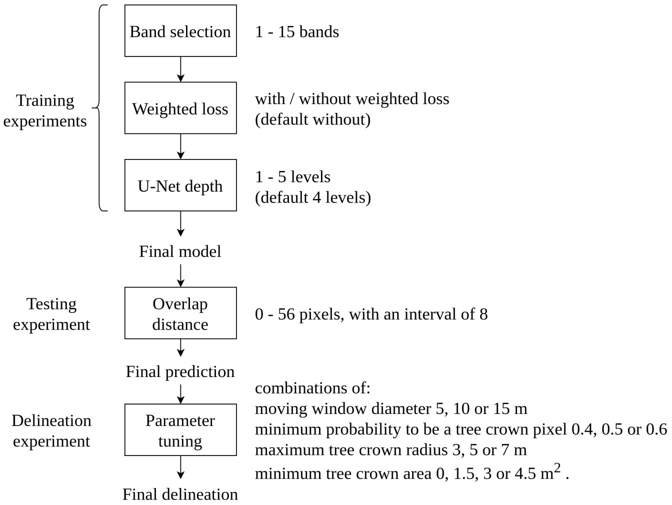
Training experiments
Band selection
1 - 15 bands
Weighted loss
with / without weighted loss (default without)
U-Net depth
1 - 5 levels (default 4 levels)
Final model
Testing experiment
Overlap distance
0 - 56 pixels, with an interval of 8
Final prediction
Delineation experiment
Parameter tuning
combinations of:
moving window diameter 5, 10 or 15 m
minimum probability to be a tree crown pixel 0.4, 0.5 or 0.6
maximum tree crown radius 3, 5 or 7 m
minimum tree crown area 0, 1.5, 3 or 4.5 m^2 .
Final delineation

Fig. 2. Overview of experiments conducted sequentially in training, testing, and delineation phases. In the training phase, former experiments used the default setting from latter experiments and latter experiments used the determined setting from former experimental results. Testing experiment used the final model from training experiment to produce final prediction, which I then used in the delineation experiment to generate final tree crown delineation results. I experimented with all combinations of the tuning parameters.

when validation loss did not decrease 0.0001 after 5 epochs. The chosen epoch was the one with the lowest validation loss.

3.4 Training experiments

I conducted three experiments sequentially in the training phase to optimize the model: band selection, weighted loss, and U-Net depth. Former experiments used the default setting from latter experiments and latter experiments used the determined setting from former experimental results (Fig. 2).

3.4.1 Band selection: High dimensionality of the input data can lead to reduced performance due to redundant learning [6]. Dimensionality reduction methods have been often applied to deal with this problem [36]. One commonly used method is the Principal Component Analysis (PCA), but it chooses bands disregarding the spatial relationship [37]. Instead of applying a specific dimensionality reduction method, studies such as [23], [24], [38] simply experimented with various combinations of spectral bands.

I decided to use the concept of forward feature selection [39] to select bands in this study. Starting from one band, I trained 15 models and selected the band with the highest validation accuracy. Then I trained 14 models using this chosen band combined with the other bands. I repeated this process until all 15 bands were chosen. For performance comparison, I also trained a baseline model on only RGB bands.

3.4.2 Weighted loss: To separate touching objects, weighted categorical cross entropy loss function was used after a soft-max output activation function [19]. By visual inspection, training and testing images in this study showed touching and overlapping tree crowns in many tiles. Hence, I tested if implementing weighted loss could improve the model.

A soft-max function produces two output numbers per pixel in [19]: one is the probability of the pixel being a tree crown pixel and the other is the probability of it being a non-crown pixel. The sigmoid function used in this study generates only one output number per pixel, indicating the probability of it being a tree crown pixel. Thus, I used binary cross entropy loss function instead of the categorical cross entropy loss function.

From [19] and [40] I derived the Weighted Binary Cross Entropy (WBCE) loss function (Equation 1), where N is total number of pixels, w is weight, y is true value, and $\hat{y}$ is predicted value. I produced a weight tile for each target tile (Appendix Fig. A.1) according to [19] (Equation 2), where cw is class weight to deal with imbalance in the number of tree crown and non-crown pixels, w_0 and σ are constants set to respectively 10 and 5 pixels, $d1$ and $d2$ are the distances (in pixel) to the nearest and the second nearest tree crown pixel. These distances are both 0 when i is a tree crown pixel. I calculated class weight with Equation 3, where n_j is the number of pixels having the value of j and the $max()$ function outputs the larger of the two input numbers. j equals 1 if pixel i is a tree crown pixel and 0 if pixel i is a non-crown pixel (Equation 4).

$$WBCE = -\frac{1}{N}\sum_{i=1}^{N} w_i[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)] \quad (1)$$

$$w_i = cw_i + w_0 * exp\left(-\frac{(d1_i + d2_i)^2}{2\sigma^2}\right) \quad (2)$$

$$cw_i = \frac{\frac{1}{n_j}}{max(\frac{1}{n_0}, \frac{1}{n_1})} \quad (3)$$

$$j = \begin{cases} 1, & \text{if pixel } i \text{ is a tree crown pixel} \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

The modified U-Net model with weighted loss implementation requires weight tiles. Thus, I could not use it directly for prediction because weight tiles are derived from target tiles and no target tiles are available in real prediction application. However, they are only involved in the WBCE calculation, which happens after model parameters are updated. Hence, another U-Net model with exactly the same architecture, but without the weighted loss implementation, can be correctly loaded with trained model parameters and used for prediction.

3.4.3 U-Net depth: ITCD training data size is usually much smaller than that in computer science [6]. Deeper models posses more parameters and are prone to overfit when the number of data is small [4]. Moreover, a lighter model requires less computational resources and consumes less energy [6]. Hence, I experimented on U-Nets with one to five levels to determine which number of level had the highest validation accuracy. The more levels a U-Net has, the deeper is the U-shape in its model architecture (Appendix Fig. A.2).

3.5 Testing experiment: overlap distance

I reassembled tile prediction results of 128 x 128 pixels produced by the final model (Fig. 2) and cropped them to their original input extent. Tile borders were however clearly visible. Therefore, I used overlap-tile strategy as explained in [19] to compensate for lower prediction quality at image borders [3], [41]. It works by creating prediction tiles that partly overlap and using only the central part of the prediction tiles to construct the final prediction.

I noticed from [41], [42], [43] that the choice of overlap distance varied with different use cases and image resolutions and there was no general suggestion on which distance to use. Thus, I inspected the test accuracy of a random testing image with an overlap distance from zero to 56 pixels, with an interval of eight, to determine the optimal value.

3.6 Delineation experiment: parameter tuning

The output of U-Net is a probability map. To delineate tree crown pixels into individual tree crowns, post-processing of the model prediction is needed [7]. I first normalized prediction results to the range of 0 and 1. Then I applied a filter with a moving window to find local maxima [44]. Further, I applied a region growing method [45] to delineate individual tree crowns. Finally, I removed tree crowns smaller than a specific size that may be hard to be confidently annotated as tree crowns by visual inspection of the images [42], [46].

The authors of the benchmark dataset paper [8] published the companion R *NeonTreeEvaluation* package for evaluating delineation results [18]. I adopted the default Intersection of Union (IoU) value of 0.4 and calculated F1-score from the evaluation output precision and recall (Equations 5 to 8). Precision represents the ratio of predictions matching an annotation and recall means the ratio of annotations correctly predicted.

$$IoU = \frac{Prediction \cap Truth}{Prediction \cup Truth} \tag{5}$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{6}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{7}$$

$$F1\text{-}score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{8}$$

To find the best delineation result based on F1-score, I conducted a grid search to tune parameters using 19 randomly selected testing files (Appendix Table A.2). Parameter combinations tested were: moving window diameter 5, 10 or 15 m; minimum probability to be considered as a tree crown pixel 0.4, 0.5 or 0.6; maximum tree crown radius 3, 5 or 7 m; and minimum tree crown area 0, 1.5, 3 or 4.5 m^2 (Fig. 2).

4. Results

4.1 Training experiments

The forward feature selection showed that the model with eight bands including CHM, band 113, ARVI, NDVI, GNDVI, band 55, PRI, and band 11 had the highest validation accuracy of 0.8764 (Fig. 3). In comparison, the baseline model trained only on RGB bands had a validation accuracy of 0.7277. Model performance tended to improve with increasing number of bands toward eight, but tended to decrease afterwards.

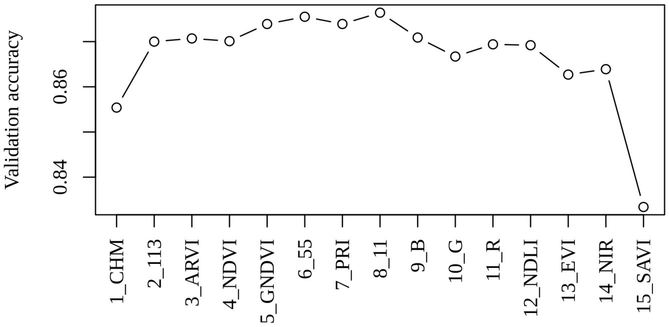

Fig. 3. Band selection result from forward feature selection. The X axis shows the consecutive order of bands selected. For example, the first selected band was CHM, which achieved the highest validation accuracy among the 15 one-band models. The second selected band was band 113, which achieved the highest validation accuracy among the 14 two-band models. Full band names are listed in Table 2. Accuracy peaked with eight bands, but the difference among two to 14 bands was marginal.

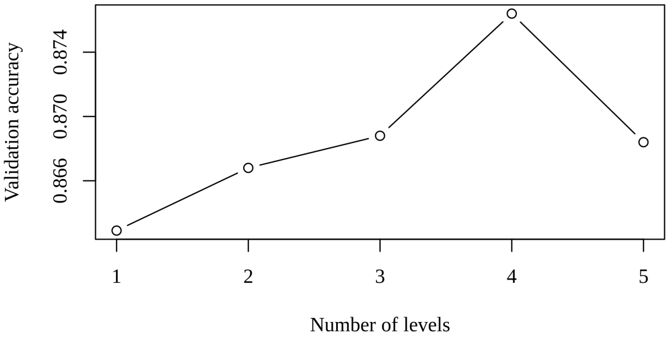

Fig. 4. Effect of U-Net depth on model validation accuracy. The more levels a U-Net has, the deeper is the U-shape in the model architecture (Appendix Fig. A.2). Validation accuracy peaked with four levels, but the difference was negligible.

The difference in validation accuracy among two to 14 bands was however marginal. Appendix Fig. A.3 shows the training history of this eight-band model.

The weighted eight-band model had a validation accuracy of 0.8539, which was lower than the one without weighted loss.

The eight-band model without weighted loss peaked in validation accuracy when the depth of U-Net was four levels (Fig. 4). The difference in validation accuracy across different levels was however negligible.

As a result, I used the eight-band model without weighted loss and with a U-Net depth of four levels for the following testing experiment.

4.2 Testing experiment: overlap distance

Larger overlap distance tended to achieve higher test accuracy (Fig. 5). Appendix Fig. A.4 visualizes the prediction results constructed from tiles with different overlap distances and shows that borders are less apparent as overlap distance increases.

I decided to use an overlap distance of 24 pixels instead of 56 to balance performance and processing time. For every 400 x 400 pixels testing file, overlapping 56 pixels would generate 625 tiles, but only 25 tiles for 24 pixels.

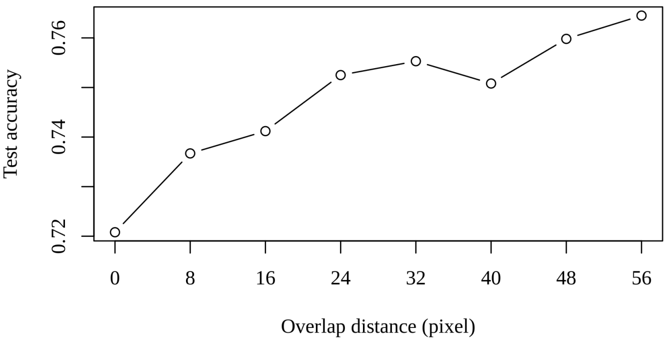

Fig. 5. Effect of overlap distance of the U-Net prediction result tiles in pixels on test accuracy. The results were based on a single test file. It showed a clear trend of accuracy improvement with increasing overlap distance.

4.3 Delineation experiment: parameter tuning

Grid search of delineation parameters revealed that the combination of moving window diameter 5 m, minimum probability to be a tree crown 0.5, maximum tree crown radius 7 m, and minimum tree crown area 3 m^2 achieved the highest F1-score of 0.3760 (precision 0.3528 and recall 0.4025). As a comparison, the minimum F1-score among the parameter combinations was much lower with 0.1951.

4.4 Evaluation

Fig. 6 visualizes some of the final delineation results. Visual inspection suggested isolated trees of middle size were delineated the best (Fig. 6a, b). Closely positioned tree crowns were not separated properly (Fig. 6a, c, d). While large tree crowns were divided into multiple segments (Fig. 6a), crowns too small were not identified (Fig. 6c, d). The U-Net did not predict shadows as tree crowns and distortion in images did not seem to affect the prediction performance much (Fig. 6b, c). There were cases where predicted tree crowns did not match annotations (Fig. 6a, c, d).

The results achieved an overall precision of 0.3634, recall of 0.4222, and F1-score of 0.3906. On the site level, the performance varied (Fig. 7). Precision at all sites were lower than 50%, meaning that only less than half of the predictions matched an annotation. Highest recall was achieved at the site SJER, with more than 60% of the annotations correctly predicted. Worst performance was observed at the sites ABBY and NIWO, where both precision and recall were nearly zero.

5. Discussion

This study presented a workflow training multi-band CNNs using data from multiple sources for delineating individual tree crowns. The final eight-band model without weighted loss having a depth of four levels reached a validation accuracy of 0.8764, outperforming that of a model using only the RGB bands by 0.15. With an overlap distance of 24 pixels across the prediction tiles and the best combination of tuning parameters, the final delineation result achieved an overall precision of 0.3634, recall of 0.4222, and F1-score of 0.3906.

5.1 Significance of this study

This study contributed to addressing data fusion from multiple sources, one of the identified research gaps of CNNs in ITCD [6], [7]. The proposed methods of training CNNs with any number of bands may facilitate future ITCD applications to more easily explore the possibilities of data fusion.

Most deep learning research was conducted using the Python programming language [3] and the Python *TensorFlow* package was the most popular deep learning framework [47]. However, I was not able to find an implementation of the U-Net pixel-wise weighted loss function in *TensorFlow*. This was probably because U-Net was first developed with a framework that later evolved into the Python *Pytorch* package [19], [48]. My translation of the codes from [49] into R did not work out. [50] tried to recreate it without success. [51] and [42] implemented a loss function different from that of the U-Net. The implementation of a close variation of the U-Net weighted loss function in R and in the *TensorFlow* framework in this study may open up new research opportunities for these communities.

5.2 Experiments and performance

In the band selection experiment, the performance decreased as the number of bands increased after eight bands, showing redundant learning. The forward feature selection algorithm was simple to implement. However, as the number of bands or training tiles increase, this method may lead to very long processing time, reducing its feasibility. Conducting preliminary experiments on only part of the available training tiles before training on the selected bands with all training tiles may be a compromise reducing dimensionality with this method. Also, one could choose to stop the forward feature selection process early when validation accuracy did not improve a certain degree after adding some bands.

The implementation of weighted loss did not improve the validation accuracy of the U-Net model. A possible explanation is that the rectangular annotations did not precisely reflect the true tree crowns at the pixel level and were partly overlapping (Fig. 6c, d). Hence the use of weighted loss did not bring an advantage and the prediction results were not able to separate touching tree crowns. Future research may train models on annotations of better quality or explore different weighted loss calculation methods and annotation strategies [51] to address the touching tree issue. This could potentially improve ITCD in tropical areas that are of high conservation value, especially in the face of the global anthropogenic climate change.

Since training and testing experiments did not always show great performance enhancement with increasing complexity of the processes, it might be useful to analyze cost (annotation work, computation time and energy consumption) and performance (precision, recall) trade-off [5]. As more and more ITCD methods were published, it would also be practical to investigate the trade-off among different methods [7].

While the U-Net prediction results matched visually well to the annotations and were not affected by shadows or distortions, delineation results of especially large and small

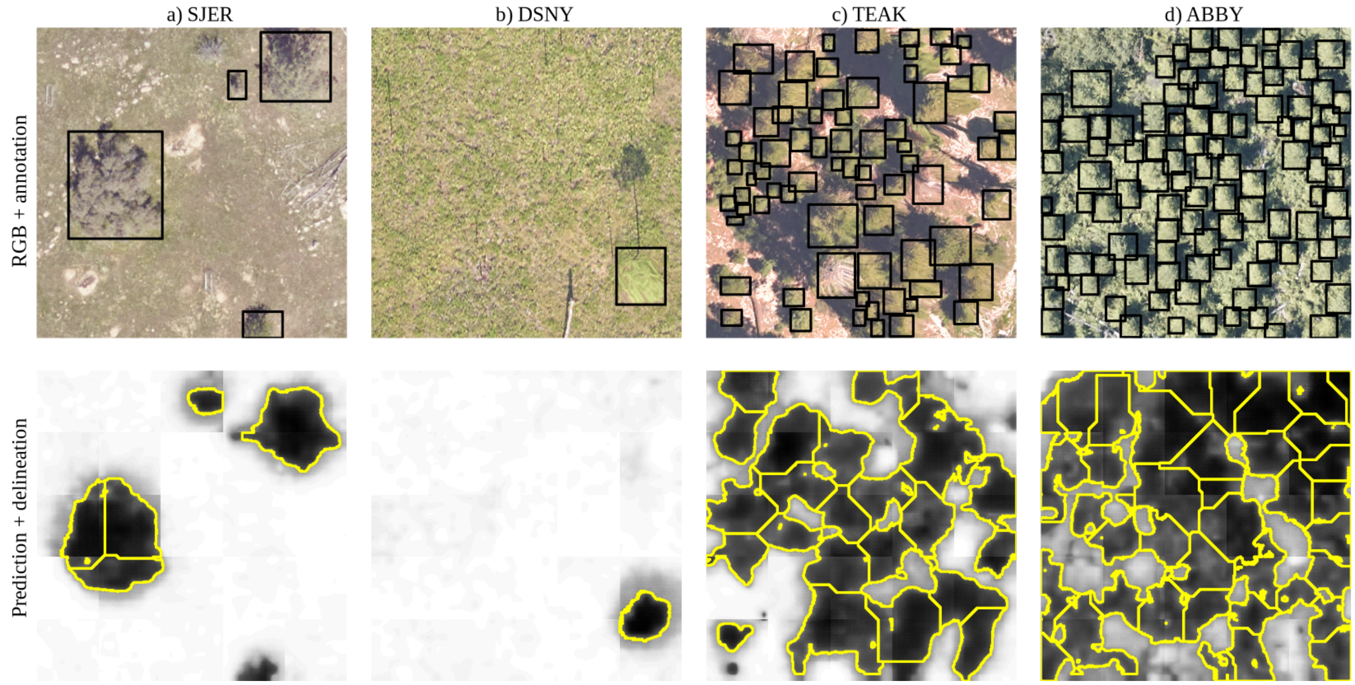

Fig. 6. Delineation result of four exemplary test files. The upper row shows the RGB image with manual annotation bounding boxes. The bottom row shows tree crown delineations (yellow) and U-Net prediction result, where darker color represents higher probability of being a tree crown pixel. Full site names are listed in Appendix Table A.1. a) The largest tree crown in the image was delineated as several tree crowns. A small tree crown at the bottom was not delineated, although it was predicted successfully. At the top of the image, two tree crowns were not separated properly and an object was falsely predicted as a tree crown. b) and c) Distortion in the RGB images due to georectification with LiDAR data and shadow did not seem to affect the U-Net prediction. c) and d) Touching tree crowns were not delineated properly and small tree crowns were not identified. False predictions are present at the top left corner of both files. Note that imprecise rectangular tree crown annotations and partially overlapping annotations may not be ideal for training the U-Net.

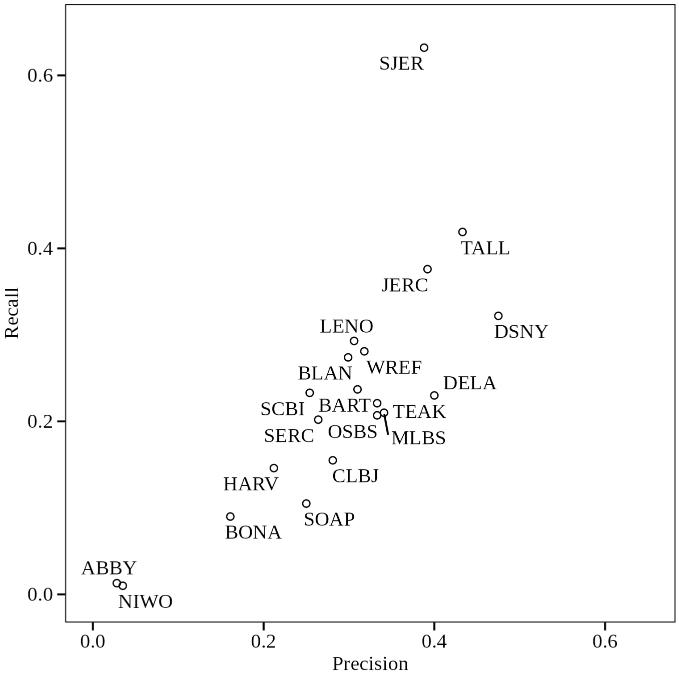

Fig. 7. Final delineation result metrics per site of 186 test files in this study. The highest precision (ratio of predictions matching an annotation) of nearly 0.5 was obtained at site DSNY and the highest recall (ratio of annotations correctly predicted) of over 0.6 was achieved at the site SJER. Lowest performance was observed at the sites NIWO and ABBY with both metrics close to 0. Full site names are listed in Appendix Table A.1.

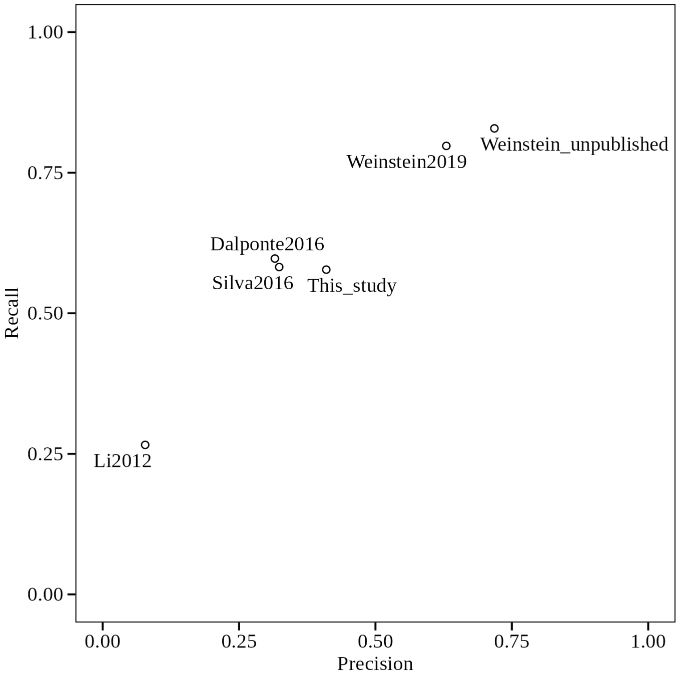

Fig. 8. Comparison of overall performance with other submitted results based on 63 common test files. The two deep learning results Weinstein_unpublished and Weinstein2019 performed the best (precision over 0.6 and recall over 0.8). This study and the two traditional image processing results Dalponte2016 and Silva2016 (precision about 0.4 and recall about 0.6) followed. The result Li2012 was the least favorable (precision and recall about 0.2).

TABLE 3
SUMMARY OF OTHER DELINEATION RESULTS. I kept the result names the same as the file names in the source [52] for traceability.

Result name	Data source	ITCD method
Dalponte2016	LiDAR (CHM)	[45], Image processing
Li2012	LiDAR	[53], Image processing
Silva2016	LiDAR (CHM)	[54], Image processing
Weinstein2019	RGB	[56], Deep learning
Weinstein_ unpublished	RGB	[56], Deep learning

tree crowns were not optimal. A clear trend of decreasing delineation performance with increasing tree crown density per site is shown in Appendix Fig. A.5. The difference in tree crown density could explain the superior performance of this study at the site SJER and the very poor performance at the sites ABBY and NIWO. Future research may further explore delineation strategies, possibly using one set of parameters on images characterizing larger and more isolated tree crowns (Fig. 6a, b), and another for smaller and more densely positioned tree crowns (Fig. 6c, d).

5.3 Comparison with other results

A benchmark dataset is a fundamental basis for comparison across algorithms. I compared the best result of this study with other delineation results provided by the authors of the benchmark dataset paper [8] available at [52] (Table 3). To make results comparable, I assessed only all of the 63 common testing files (Appendix Table A.2) and set the IoU to 0.4.

The results Dalponte2016, Li2012, and Silva2016 were obtained by using a now deprecated function in the R *lidR* package on the LiDAR data. For the results Dalponte2016 and Silva2016, the LiDAR data were only used for deriving CHM. The methods [45], [53], [54] can be classified as region growing methods in the traditional image processing methods of ITCD according to [7], [55]. The method [53] was developed to delineate mixed conifer forests based on horizontal spacing of individual trees. [54] and [45] used the height information to delineate trees. While the former focused on a single conifer species at the site JERC, the latter was designed for various coniferous and broad-leaf trees. Note that [45] is the same region growing method I used in the delineation experiment.

The results Weinstein2019 and Weinstein_unpublished were generated by different versions of the Python *DeepForest* package. The package was based on the methods proposed in [56], which can be classified as a deep learning ITCD method according to [7]. The object detection CNN model needed RGB images as input. It was first trained with pre-trained weights on a large number of annotations generated with the [54] method and then retrained on some manual annotations [56].

Both *DeepForest* results outperformed other results based on 63 common testing files, reaching more than 0.6 in precision and 0.8 in recall (Fig. 8). This study achieved a slightly higher precision than the results Dalponte2016 and Silva2016, and reached about the same recall as them. The results Li2012 were the least favorable.

The superior performance of the method [56] over this study was probably due to the use of another CNN architecture, pre-trained weights, millions of automatically generated tree crown annotations, and thousands of manual annotations. This study trained on much fewer data from scratch, but still achieved relatively good results.

5.4 Limitation and application

This study had a major limitation of using just 1% of the available tiles for training. However, even the model trained on only 100 tiles, it performed relatively well. More training data or data augmentation may contribute to enhanced prediction performance of the U-Net. Further research should address this problem with high priority by, for example, training with larger GPU memory or exploring how to most efficiently train models using big data consecutively in smaller portions.

Another limitation is sample representativeness when selecting training tiles and testing files for the experiments. Strategies of selecting representative samples [6] and the use of cross-validation in training [46] may be a future research direction that would increase the robustness of the model.

The model proposed in this study was trained on data across various forest conditions (Appendix Table A.2). The transferability of this model would still need further investigation. It is worth noting, however, that CHM (LiDAR) and hyperspectral data at a resolution of 1 m may not yet be readily available at any place. Data of even coarser resolution may make it more challenging for delineating small tree crowns.

6. CONCLUSION

In summary, this study found that the U-Net model performed better in the ITCD benchmark dataset when trained on data fused from multiple sources than only the usual RGB bands. The model was able to predict tree crown pixels and not influenced by torching images or shadows. The attempt to segment overlapping tree crowns was not successful, but training with more data with higher annotation quality or developing better delineation strategies may improve the results.

The proposed workflow may pave the way for future Earth observation researchers to apply data fusion to CNNs. Furthermore, the novel implementation of weighted loss in the *TensorFlow* framework in R may open its applications in other fields for segmenting touching or overlapping objects for new communities. However, future research should tackle the memory problem and enhance the robustness of the model.

7. ACKNOWLEDGMENT

I thank Dirk Zeuss and Lisa Bald sincerely for their valuable guidance and insightful suggestions throughout this study and for the provision of hardware. I also express my deep gratitude to Christian Mestre Runge for the helpful feedback in making the codes more readable and reproducible. My genuine gratefulness goes to Henri Dümpelmann, I-Ying Hsu, Kai Richter, and Konstantin Engelmayer for reviewing the manuscript to make this work more complete and understandable. Last but not least, I truly appreciate the profound support of my family, friends, and the society of Marburg over the course of my university studies.

8. Licence

References

[1] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," 2015. [Online]. Available: https://arxiv.org/abs/1511.08458v2

[2] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," 2018. [Online]. Available: https://arxiv.org/abs/1603.07285v2

[3] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, "Review on Convolutional Neural Networks (CNN) in vegetation remote sensing," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 173, pp. 24–49, Mar. 2021.

[4] T. Hoeser, F. Bachofer, and C. Kuenzer, "Object Detection and Image Segmentation with Deep Learning on Earth Observation Data: A Review - Part II: Applications," *Remote Sensing*, vol. 12, no. 18, p. 3053, Sep. 2020.

[5] T. Hoeser and C. Kuenzer, "Object Detection and Image Segmentation with Deep Learning on Earth Observation Data: A Review - Part I: Evolution and Recent Trends," *Remote Sensing*, vol. 12, no. 10, p. 1667, May 2020.

[6] H. Zhao, J. Morgenroth, G. Pearse, and J. Schindler, "A Systematic Review of Individual Tree Crown Detection and Delineation with Convolutional Neural Networks (CNN)," *Current Forestry Reports*, vol. 9, no. 3, pp. 149–170, Apr. 2023.

[7] J. Zheng, S. Yuan, W. Li, H. Fu, and L. Yu, "A review of individual tree crown detection and delineation from optical remote sensing images," Oct. 2023. [Online]. Available: https://arxiv.org/abs/2310.13481v1

[8] B. G. Weinstein, S. J. Graves, S. Marconi, A. Singh, A. Zare, D. Stewart, S. A. Bohlman, and E. P. White, "A benchmark dataset for canopy crown detection and delineation in co-registered airborne RGB, LiDAR and hyperspectral imagery from the National Ecological Observation Network," *PLOS Computational Biology*, vol. 17, no. 7, p. e1009180, Jul. 2021.

[9] B. Weinstein, S. Marconi, and E. White, "Data for the NeonTreeEvaluation Benchmark," 2022. [Online]. Available: https://doi.org/10.5281/zenodo.5914554

[10] B. Weinstein, "NeonTreeEvaluation," 2021. [Online]. Available: https://github.com/weecology/NeonTreeEvaluation/tree/1.8.0

[11] B. G. Weinstein, S. Marconi, S. A. Bohlman, A. Zare, and E. P. White, "Cross-site learning in deep learning rgb tree crown detection," *Ecological Informatics*, vol. 56, p. 101061, Mar. 2020.

[12] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2024, R version 4.3.3. [Online]. Available: https://www.R-project.org

[13] J. J. Allaire and Y. Tang, *tensorflow: R Interface to 'TensorFlow'*, 2024, R package version 2.16.0. [Online]. Available: https://CRAN.R-project.org/package=tensorflow

[14] J. J. Allaire and F. Chollet, *keras: R Interface to 'Keras'*, 2024, R package version 2.15.0. [Online]. Available: https://CRAN.R-project.org/package=keras

[15] R. J. Hijmans, *terra: Spatial Data Analysis*, 2023, R package version 1.7.65. [Online]. Available: https://CRAN.R-project.org/package=terra

[16] J.-R. Roussel, D. Auty, N. C. Coops, P. Tompalski, T. R. H. Goodbody, A. S. Meador, J.-F. Bourdon, F. de Boissieu, and A. Achim, "lidR: An R package for analysis of Airborne Laser Scanning (ALS) data," *Remote Sensing of Environment*, vol. 251, p. 112061, 2020.

[17] J.-R. Roussel and D. Auty, *Airborne LiDAR Data Manipulation and Visualization for Forestry Applications*, 2024, R package version 4.1.1. [Online]. Available: https://cran.r-project.org/package=lidR

[18] B. Weinstein, "NeonTreeEvaluation_package," 2020. [Online]. Available: https://github.com/weecology/NeonTreeEvaluation_package/tree/c4d99533e5dfb80f9e0ec32e66623001ea139bbd

[19] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional Networks for Biomedical Image Segmentation*. Springer International Publishing, 2015, pp. 234–241.

[20] B. Weinstein, "Information of bands used," Feb. 2024. [Online]. Available: https://github.com/weecology/NeonTreeEvaluation/issues/36#issuecomment-1937795556

[21] X. Xi, K. Xia, Y. Yang, X. Du, and H. Feng, "Evaluation of dimensionality reduction methods for individual tree crown delineation using instance segmentation network and UAV multispectral imagery in urban forest," *Computers and Electronics in Agriculture*, vol. 191, p. 106506, Dec. 2021.

[22] A. Safonova, E. Guirado, Y. Maglinets, D. Alcaraz-Segura, and S. Tabik, "Olive Tree Biovolume from UAV Multi-Resolution Image Segmentation with Mask R-CNN," *Sensors*, vol. 21, no. 5, p. 1617, Feb. 2021.

[23] I. Ulku, E. Akagunduz, and P. Ghamisi, "Deep Semantic Segmentation of Trees Using Multispectral Images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 7589–7604, 2022.

[24] J. Maschler, C. Atzberger, and M. Immitzer, "Individual Tree Crown Segmentation and Classification of 13 Tree Species Using Airborne Hyperspectral Data," *Remote Sensing*, vol. 10, no. 8, p. 1218, Aug. 2018.

[25] D. Hulslander, "NEON NDVI, EVI, Canopy Xanthophyll Cycle, and Canopy Lignin Algorithm Theoretical Basis Document," National Ecological Observatory Network, Tech. Rep. NEON.DOC.002391vB, 2022.

[26] Y. J. Kaufman and D. Tanre, "Atmospherically resistant vegetation index (ARVI) for EOS-MODIS," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 30, no. 2, pp. 261–270, Mar. 1992.

[27] A. R. Huete, "A soil-adjusted vegetation index (SAVI)," *Remote Sensing of Environment*, vol. 25, no. 3, pp. 295–309, Aug. 1988.

[28] A. A. Gitelson, Y. J. Kaufman, and M. N. Merzlyak, "Use of a green channel in remote sensing of global vegetation from EOS-MODIS," *Remote Sensing of Environment*, vol. 58, no. 3, pp. 289–298, Dec. 1996.

[29] J. A. Gamon, J. Peñuelas, and C. B. Field, "A Narrow-Waveband Spectral Index That Tracks Diurnal Changes in Photosynthetic Efficiency," *Remote Sensing of Environment*, vol. 41, no. 1, pp. 35–44, Jul. 1992.

[30] L. Serrano, J. Peñuelas, and S. L. Ustin, "Remote sensing of nitrogen and lignin in mediterranean vegetation from aviris data," *Remote Sensing of Environment*, vol. 81, no. 2–3, pp. 355–364, Aug. 2002.

[31] r-tensorflow, "unet," 2019. [Online]. Available: https://github.com/r-tensorflow/unet/blob/c47cf31f13050722b587a5c394d4511d8f5e50b9/R/model.R

[32] TensorFlow, "TensorFlow API documentation for module: tf.image," Jan. 2024. [Online]. Available: https://www.tensorflow.org/versions/r2.15/api_docs/python/tf/image

[33] TensorFlow, "Mixed precision," Jul. 2023. [Online]. Available: https://github.com/tensorflow/docs/blob/13e81b18095afd2fd40b0300d465e0a7b4592a5d/site/en/guide/mixed_precision.ipynb

[34] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, Jul. 2019.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," 2015.

[36] N. Audebert, B. Le Saux, and S. Lefèvre, "Deep Learning for Classification of Hyperspectral Data: A Comparative Review," 2019. [Online]. Available: https://arxiv.org/abs/1904.10674v1

[37] G. T. Miyoshi, M. d. S. Arruda, L. P. Osco, J. Marcato Junior, D. N. Gonçalves, N. N. Imai, A. M. G. Tommaselli, E. Honkavaara, and W. N. Gonçalves, "A Novel Deep Learning Method to Identify Single Tree Species in UAV-Based Hyperspectral Images," *Remote Sensing*, vol. 12, no. 8, p. 1294, Apr. 2020.

[38] S. Yao, Z. Hao, C. J. Post, E. A. Mikhailova, and L. Lin, "Individual Tree Crown Detection and Classification of Live and Dead Trees Using a Mask Region-Based Convolutional Neural Network (Mask R-CNN)," *Forests*, vol. 15, no. 11, p. 1900, Oct. 2024.

[39] A. Hennessy, K. Clarke, and M. Lewis, "Hyperspectral Classification of Plants: A Review of Waveband Selection Generalisability," *Remote Sensing*, vol. 12, no. 1, p. 113, Jan. 2020.

[40] keras, "binary_crossentropy funciton," Aug. 2023. [Online]. Available: https://github.com/keras-team/keras/blob/r2.15/keras/backend.py#L5802

[41] M. Freudenberg, N. Nölke, A. Agostini, K. Urban, F. Wörgötter, and C. Kleinn, "Large Scale Palm Tree Detection In High Resolution Satellite Images Using U-Net," *Remote Sensing*, vol. 11, no. 3, p. 312, Feb. 2019.

[42] M. Brandt, C. J. Tucker, A. Kariryaa, K. Rasmussen, C. Abel, J. Small, J. Chave, L. V. Rasmussen, P. Hiernaux, A. A. Diouf, L. Kergoat, O. Mertz, C. Igel, F. Gieseke, J. Schöning, S. Li, K. Melocik, J. Meyer, S. Sinno, E. Romero, E. Glennie, A. Montagu, M. Dendoncker, and R. Fensholt, "An unexpectedly large count of trees in the West African Sahara and Sahel," *Nature*, vol. 587, no. 7832, pp. 78–82, Oct. 2020.

[43] B. Neupane, T. Horanont, and N. D. Hung, "Deep learning based banana plant detection and counting using high-resolution red-green-blue (RGB)

images collected from unmanned aerial vehicle (UAV)," *PLOS ONE*, vol. 14, no. 10, p. e0223906, Oct. 2019.

[44] S. C. Popescu and R. H. Wynne, "Seeing the Trees in the Forest: Using Lidar and Multispectral Data Fusion with Local Filtering and Variable Window Size for Estimating Tree Height," *Photogrammetric Engineering & Remote Sensing*, vol. 70, no. 5, pp. 589–604, May 2004.

[45] M. Dalponte and D. A. Coomes, "Tree-centric mapping of forest carbon density from airborne laser scanning and hyperspectral data," *Methods in Ecology and Evolution*, vol. 7, no. 10, pp. 1236–1245, May 2016.

[46] M. Freudenberg, P. Magdon, and N. Nölke, "Individual tree crown delineation in high-resolution remote sensing images based on U-Net," *Neural Computing and Applications*, vol. 34, no. 24, pp. 22 197–22 207, Aug. 2022.

[47] G. Nguyen, S. Dlugolinsky, M. Bobák, V. Tran, Á. López García, I. Heredia, P. Malík, and L. Hluchý, "Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 77–124, Jan. 2019.

[48] facebookarchive, "caffe2," 2018. [Online]. Available: https://github.com/facebookarchive/caffe2/tree/5f7cccf14a453cb8b7d556e2d2a3fa2be699185f

[49] J. Deshpande, "Weighted Loss Functions for Instance Segmentation," Aug. 2018. [Online]. Available: https://jaidevd.com/posts/weighted-loss-functions-for-instance-segmentation/

[50] M. J. Söderberg, A. Ivarsson, and J. Stachowicz, "Recreation, U-net: Convolutional networks for biomedical image segmentation," Oct. 2020. [Online]. Available: https://github.com/jacobstac/Recreation-of-U-net-Convolutional-networks-for-biomedical-image-segmentation

[51] D. Lusk, "Understanding weight maps and label manipulation in tree detection from high-resolution orthophotos with U-Net," University of Potsdam, May 2023. [Online]. Available: https://up-rs-esp.github.io/posts/2023/05/weight-maps-label-manipulation-tree-detection-unet

[52] weecology, "NeonTreeEvaluation_analysis," Mar. 2020. [Online]. Available: https://github.com/weecology/NeonTreeEvaluation_analysis/tree/a426a1a6a621b67f11dc4e6cc46eb9df9d0fc677

[53] W. Li, Q. Guo, M. K. Jakubowski, and M. Kelly, "A New Method for Segmenting Individual Trees from the Lidar Point Cloud," *Photogrammetric Engineering & Remote Sensing*, vol. 78, no. 1, pp. 75–84, Jan. 2012.

[54] C. A. Silva, A. T. Hudak, L. A. Vierling, E. L. Loudermilk, J. J. O'Brien, J. K. Hiers, S. B. Jack, C. Gonzalez-Benecke, H. Lee, M. J. Falkowski, and A. Khosravipour, "Imputation of Individual Longleaf Pine (Pinus palustrisMill.) Tree Attributes from Field and LiDAR Data," *Canadian Journal of Remote Sensing*, vol. 42, no. 5, pp. 554–573, Jul. 2016.

[55] Y. Ke and L. J. Quackenbush, "A review of methods for automatic individual tree-crown detection and delineation from passive remote sensing," *International Journal of Remote Sensing*, vol. 32, no. 17, pp. 4725–4747, Jul. 2011.

[56] B. G. Weinstein, S. Marconi, S. Bohlman, A. Zare, and E. White, "Individual Tree-Crown Detection in RGB Imagery Using Semi-Supervised Deep Learning Neural Networks," *Remote Sensing*, vol. 11, no. 11, p. 1309, Jun. 2019.

Appendix

TABLE A.1
Full site names

Site	Site Name
ABBY	Abby Road
BART	Bartlett Experimental Forest
BLAN	Blandy Experimental Farm
BONA	Caribou-Poker Creeks Research Watershed
CLBJ	Lyndon B. Johnson National Grassland
DELA	Dead Lake
DSNY	Disney Wilderness Preserve
HARV	Harvard Forest & Quabbin Watershed
JERC	The Jones Center At Ichauway
LENO	Lenoir Landing
MLBS	Mountain Lake Biological Station
NIWO	Niwot Ridge
OSBS	Ordway-Swisher Biological Station
SCBI	Smithsonian Conservation Biology Institute
SERC	Smithsonian Environmental Research Center
SJER	San Joaquin Experimental Range
SOAP	Soaproot Saddle
TALL	Talladega National Forest
TEAK	Lower Teakettle
TOOL	Toolik Field Station
WREF	Wind River Experimental Forest

TABLE A.2
Site Metadata Analysis. The table shows, of each site in the complete file sets identified in this study, the number of training tree crowns, number of training tiles including both tree crown and non-crown pixels, number of training tiles actually used in training, number of testing tree crowns, number of testing files, number of testing files selected for parameter tuning in the delineation experiment, number of common testing files for comparison with other results, and main land cover classes. Full site names are listed in Appendix Table A.1. About 60% of the training tiles and testing files originated from the sites SJER and TEAK. The random selection of training tiles due to memory constraint and of testing files for parameter tuning also reflect the uneven distribution of data across sites. All testing files for result comparison were from the two sites. The main land covers were evergreen and deciduous forests.

Site	Training crowns	Training tiles	Used tiles	Testing crowns	Testing files	Tuning files	Common files	Land Cover
ABBY	-	-	-	160	2	1	-	Evergreen Forest, Grassland/Herbaceous, Shrub/Scrub
BART	369	44	1	93	2	-	-	Deciduous Forest, Evergreen Forest, Mixed Forest
BLAN	-	-	-	73	2	-	-	Deciduous Forest, Pasture/Hay
BONA	-	-	-	255	4	1	-	Deciduous Forest, Evergreen Forest, Mixed Forest, Woody Wetlands
CLBJ	-	-	-	116	2	1	-	Deciduous Forest, Grassland/Herbaceous
DELA	295	99	1	87	2	-	-	Evergreen Forest, Woody Wetlands
DSNY	-	-	-	87	6	-	-	Pasture/Hay, Woody Wetlands
HARV	329	92	3	171	3	-	-	Deciduous Forest, Evergreen Forest, Mixed Forest, Woody Wetlands
JERC	193	60	1	101	6	1	-	Cultivated Crops, Deciduous Forest, Evergreen Forest, Mixed Forest
LENO	554	137	1	75	2	-	-	Deciduous Forest, Woody Wetlands
MLBS	1921	358	4	481	8	1	-	Deciduous Forest
NIWO	9730	608	7	1624	11	2	-	Evergreen Forest, Grassland/Herbaceous
OSBS	1813	670	7	497	14	2	-	Emergent Herbaceous Wetlands, Evergreen Forest, Woody Wetlands
SCBI	-	-	-	73	2	1	-	Deciduous Forest, Evergreen Forest, Pasture/Hay
SERC	-	-	-	94	2	-	-	Cultivated Crops, Deciduous Forest
SJER	2585	4967	53	418	59	5	30	Evergreen Forest, Grassland/Herbaceous, Shrub/Scrub
SOAP	-	-	-	114	2	-	-	Evergreen Forest, Shrub/Scrub
TALL	-	-	-	31	1	-	-	Deciduous Forest, Evergreen Forest, Mixed Forest
TEAK	3670	2025	22	1468	51	3	33	Evergreen Forest, Shrub/Scrub
TOOL	1	1	-	-	-	-	-	Dwarf Scrub, Shrub/Scrub
WREF	-	-	-	178	5	1	-	Evergreen Forest
sum	21460	9061	100	6196	186	19	63	

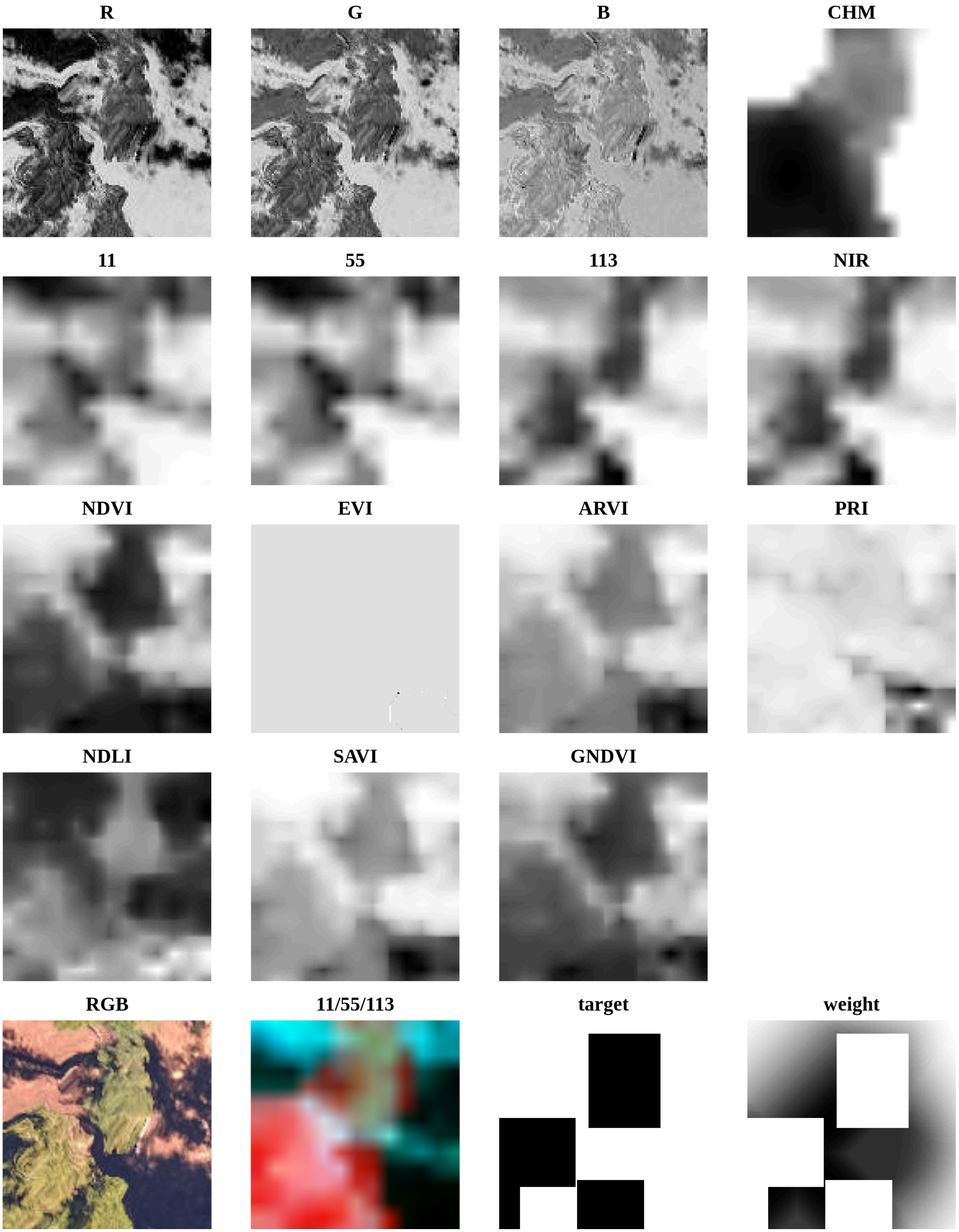

Fig. A.1. Visualization of training data for an exemplary tile. The first four rows show the 15 predictor bands used in this study, normalized to the range of 0 (white) and 1 (black). Full band names are listed in Table 2. All bands had the same resolution of 0.1 m, but bands apart from RGB were resampled from a resolution of 1 m. EVI in this tile did not show tree crown structure as other bands and may interfere model learning. Distortion due to georectification with LiDAR data are visible in the RGB image. 11/55/113 is a composite image of the three hyperspectral bands used for annotation. Target tile was derived from the manual annotations, where 0 (white) represents non-crown and 1 (black) tree crown. In the weight tile, non-crown pixels closer to tree crown has higher weight value (darker), forcing the model to learn the gaps between closely positioned tree crowns.

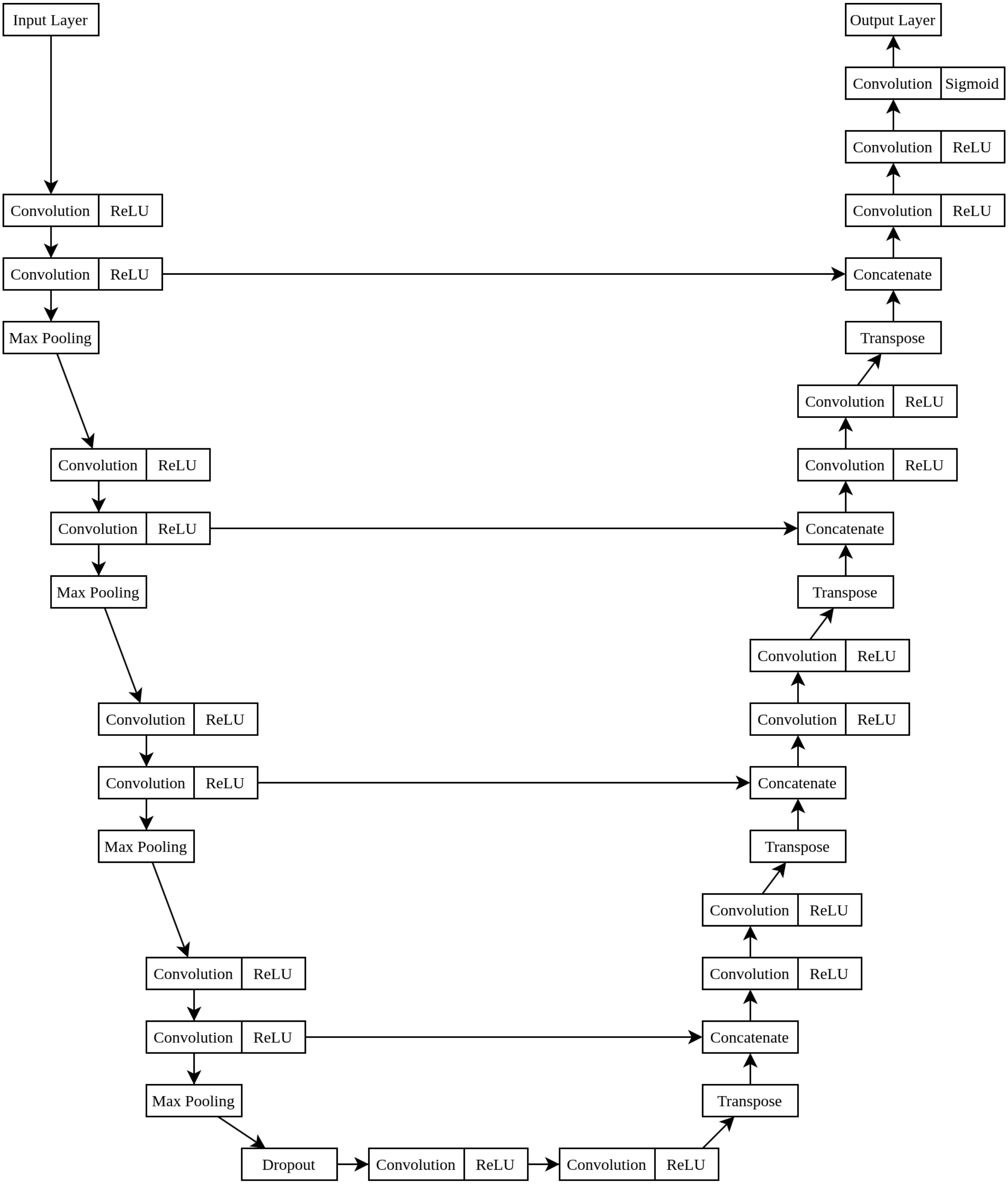

Fig. A.2. Model architecture of a U-Net with four levels used in this study. The contracting path is on the left-hand side and the expansive path is on the right-hand side. The modular design allowed easy adaptation of the model depth. ReLU: Rectified Linear Unit. Except for the input and output layer, all other layers are hidden layers of an ANN.

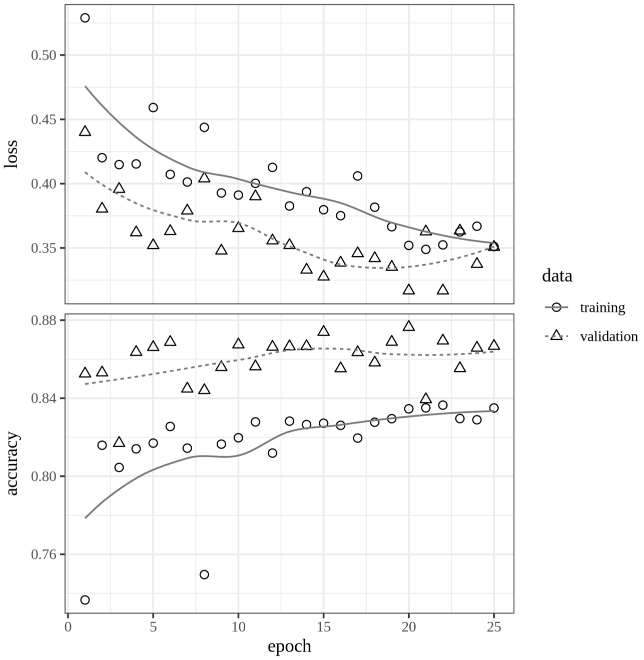

Fig. A.3. Training history of the final eight-band U-Net model without weighted loss having four levels. Training ended at epoch 25. Training and validation accuracy tended to increase, while their loss tended to decrease over the course of training. Validation accuracy was higher than training accuracy and validation loss was lower than training loss. This shows no sign of overfitting.

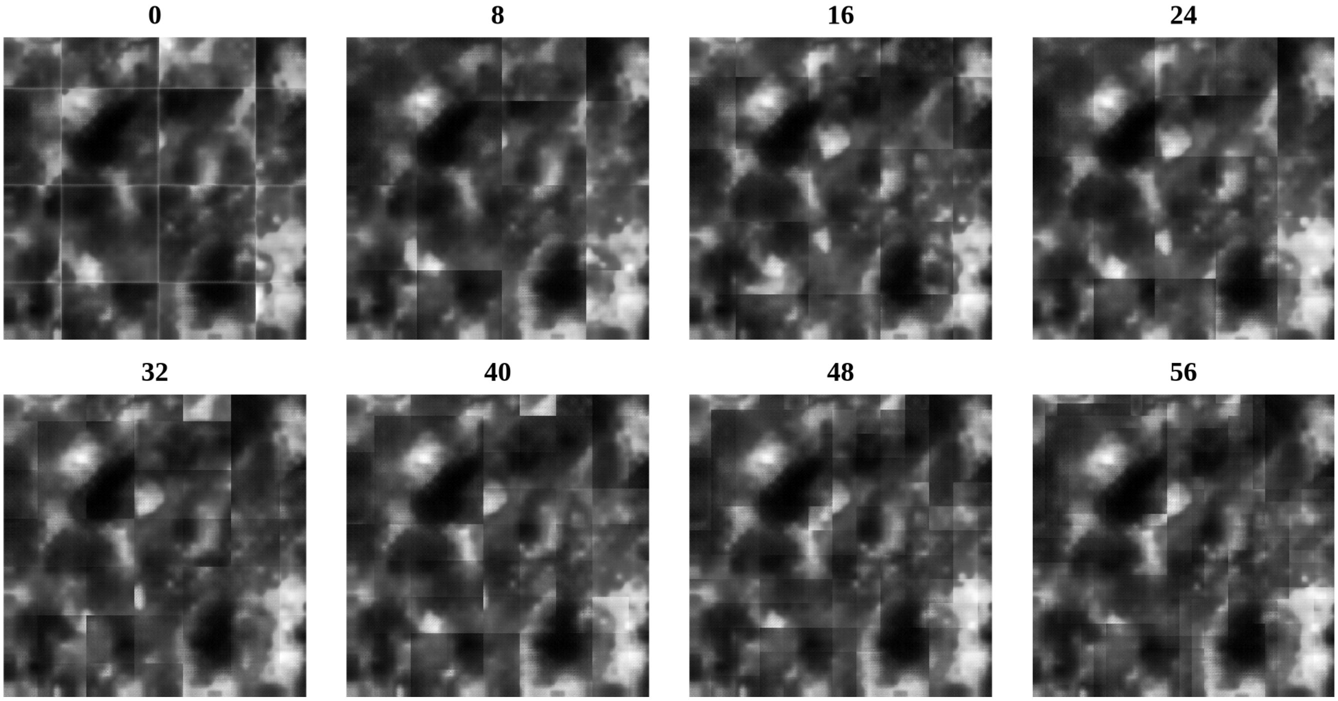

Fig. A.4. Prediction results of a testing file constructed from tiles with an overlap distance of 0 to 56 pixels with an interval of 8. Tile borders are less apparent with increasing overlap distance.

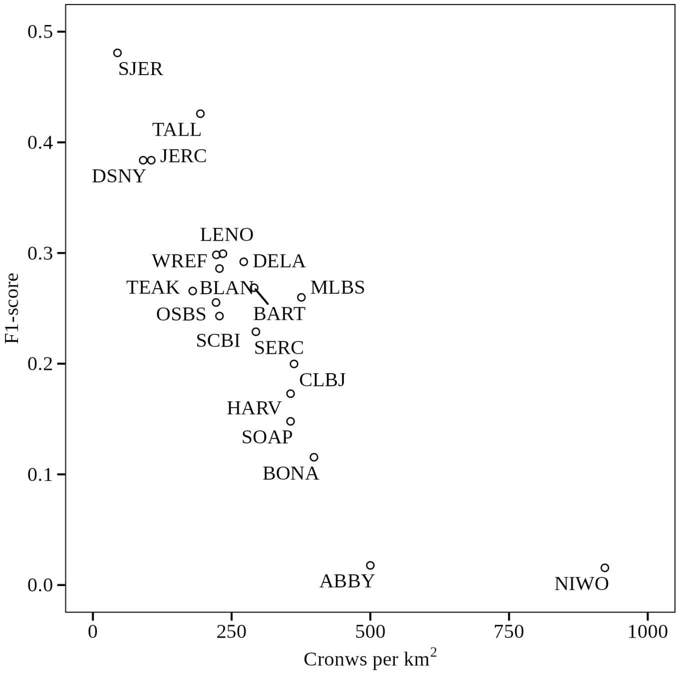

Fig. A.5. F1-score of the final 186 delineation results per site of this study versus tree crown annotation density per km^2. It showed a clear trend of decreasing performance with increasing tree crown density. Full site names are listed in Appendix Table A.1.

Diese Erklärungen sind in jedes Exemplar der Abschlussarbeit mit einzubinden

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Die Masterarbeit wurde in der jetzigen oder einer ähnlichen Form noch bei keiner anderen Hochschule eingereicht und hat noch keinen sonstigen Prüfungszwecken gedient.

Datum: 07.02.2025 Unterschrift: Chen, You-De

Einverständniserklärung zur Einsichtnahme von Abschlussarbeiten

☒ Ich bin damit einverstanden, dass meine Abschlussarbeit im Fachbereichs-/Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.

○ Ich bin <u>nicht</u> damit einverstanden, dass meine Abschlussarbeit im Fachbereichs-/Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.

Datum: 07.02.2025 Unterschrift: Chen, You-De