Centre National de la Recherche Scientifique (CNRS)
Unité mixte de recherche (UMR) 6554
Littoral – Environnement – Télédétection – Géomatique (LETG)

Internship report

UAV-based biomass estimation
–
Regression of vegetation parameters in sparse training data environments

written by
Felix Kröber

supervised by
Thomas Corpetti, Guglielmo Fernandez Garcia

Rennes, September 2022

Content

Figures & Tables

Abbreviations

AE	Autoencoder
CI	Confidence Interval
CNN	Convolutional Neural Network
K	Thousand(s)
KL	Kullback Leibler
L1	L-1 norm (MAE)
L2	L-2 norm (MSE)
LR	Learning Rate
M	Million(s)
MAE	Mean Absolute Error
MSE	Mean Squared Error
NN	Nearest Neighbour
NDVI	Normalised Difference Vegetation Index
PCA	Principal Component Analysis
RF	Random Forest
RMSE	Root Mean Square Error
t-SNE	t-Distributed Stochastic Neighbour Embedding
SOTA	State-of-the-Art
UAV	Unmanned Aerial Vehicle

1 Introduction – Problem setting & objectives

Analyses of vegetation at the field scale require high spatial and flexible temporal resolutions. Therefore, Unmanned Aerial Vehicles (UAVs) have proven to be suitable platforms to perform data collection for subsequent analyses of a wide range of potential variables of interest [1–3]. Compared to time- and cost-intensive in-situ-measurements, such remote sensing based explorations offer the advantage of being able to automate the data acquisition process in an efficient and non-destructive manner.

The range of existing UAV-based approaches for estimating biomass related variables is wide and primarily depends on the specific variable of interest. Fractional vegetation cover regression, for example, can be carried out accurately relying on well-known indices such as NDVI, which used as a proxies show high correlations with the actual variable of interest [4,5]. For vegetation height and volumes, on the other hand, more complex approaches – often involving the calculation of 3D information based on structure from motion – are commonly used [6–11]. While deep learning based classification approaches have been tested regularly on UAV data, solving regression tasks via CNNs is still underrepresented [12].

Approaches of this kind require more complex models from the field of computer vision and are thus usually based on different forms of machine learning. The limited availability of extensive, high-quality training data is a basic problem that makes the actually necessary use of complex models prone to overfitting, i.e. poor generalisation capability [13]. One fundamental problem is the insufficiently understood relationship between model complexity and expressivity of a network [14]. Very simple networks with few parameters can already perfectly fit a data set. Conversely, complex models can still have good generalisability, depending on the architecture. In practice, apart from the simplifying assumption that the expressive power of a network tends to increase with the number of free parameters, the application of various empirically advantageous regularisation techniques [15] has therefore become established.

Investigating the discrepancy of large required model complexity vs. small amount of data in the context of an automated derivation of vegetation parameters is the focus of the current work. Aiming at solving the regression problem of biomass variables, three fundamentally different approaches to tackle overfitting and achieve generalisability are compared.

2 Data basis

2.1 Project setting

The data was acquired as part of the interdisciplinary project “Positive Plant-Plant interactions and spatial Patterns in Pyrenean Post-mine tailings (SixP)” funded by the French National Research Agency. This project focuses on ecological analyses of plants growing in metal-rich soils in former mining areas [16]. One partial contribution to this concerns the development of innovative computer vision approaches to identify plant species and to map the general vegetation structure in terms of biomass variables.

UAV and in-situ-measurements were taken for 5 sites in different altitude zones in the Pyrenees in the period spring to summer 2021. All sites are located within a 5 km radius and have extensions of less than 100 m in each direction. UAVs were equipped with R, G, B, NIR sensors and the obtained images post-processed to derive at orthophotos with a resolution of either 2 or 3 mm. In-situ-measurements were taken for the three variables “fractional vegetation coverage”, “vegetation volume” and “vegetation height”. A spatially stratified random sampling design with about 40 measurements per side was applied leading to a total of 198 measurements across all sites. For each point, fractional vegetation coverage, vegetation volume and plant heights were recorded based on a 25 x 25 cm square representative for the 1 x 1 m area centered around the point

2.2 Data pre-processing

To derive at a homogenous data set across sites, the UAV data was resampled to 3 mm resolution and contrast stretching applied for each site individually. The resulting images were saved in 16-bit format. Fig. 1 shows a comparison of the spectra between the sites before the application of these corrections and afterwards.

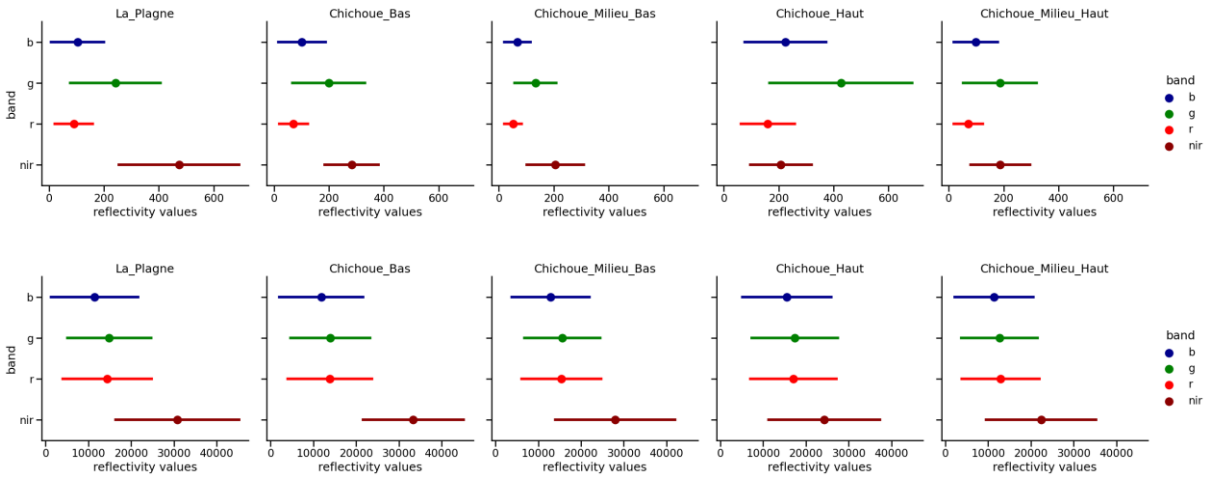

Figure 1: Mean values & standard deviations of the reflectance per band prior to (top) and after (bottom) applying contrast-stretching

Two different sub data sets were created out of the UAV images. First, the UAV images were cropped to the in-situ-measurement geolocations generating tiles with a 1 x 1 m extent centred around each point measurement. Subsequently, each tile – plotted as R,G,B and false-colour NIR,G,B – was assessed visually to exclude unreliable tiles with obvious deviations from recorded in-situ-measurements. After excluding these measurements, as well as those for which there is no complete overlap through the UAV images, a set of 160 measurements and associated UAV tiles remained for further analysis. Statistics on the distribution of the values and the correlation of variables can be found in Figs. 2 & 3. It is evident that the vegetation variables vary not only within but also across sites. La Plagne representing the lowest altitude site shows the strongest vegetation whereas Chichoue Milieu Bas, for example, shows only sparse vegetation.

In addition to this dataset directly aligned to the geolocation of the measurements, a second dataset was generated by tiling the complete UAV images. With an overlap of 50 %, this results in the numbers of tiles per site listed in Table 2. Since no direct measurements of the vegetation parameters are available for these tiles, they are referred to as unlabelled tiles subsequently.

Table 1: Number of unlabelled tiles per site

site name	site abbreviation	number of tiles
Chichoue Bas	CB	7786
Chichoue Milieu Bas	CMB	3603
Chichoue Haut	CH	6454
Chichoue Milieu Haut	CMH	9167
La Plagne	P	9625

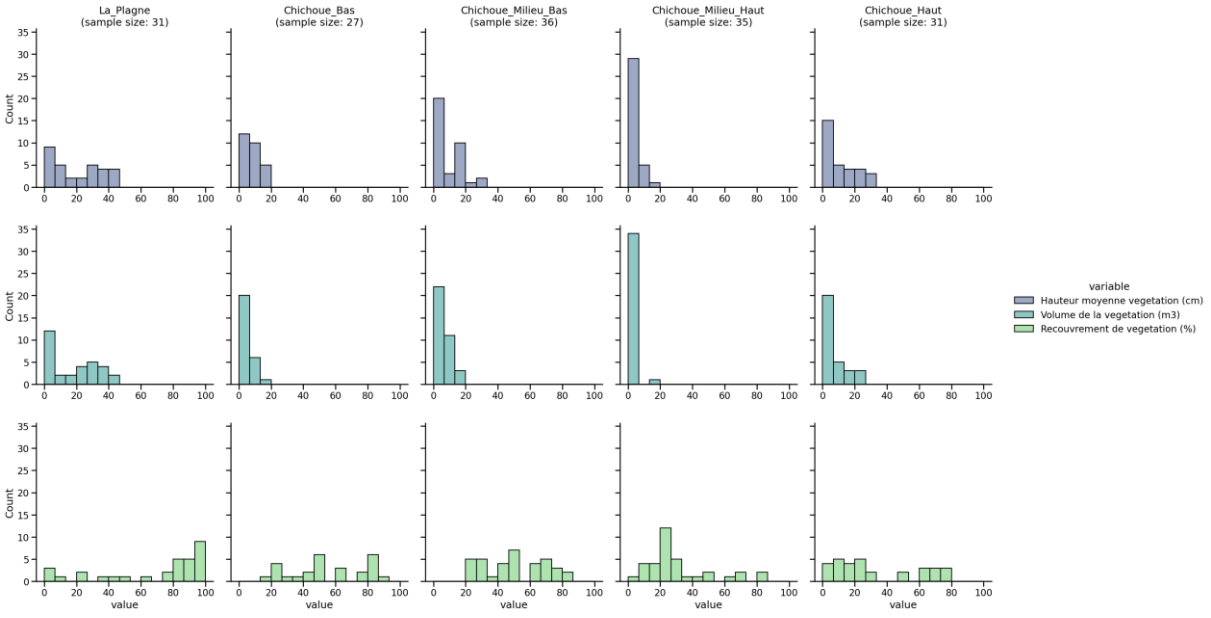

Figure 2: Frequency distributions for each variable across sites

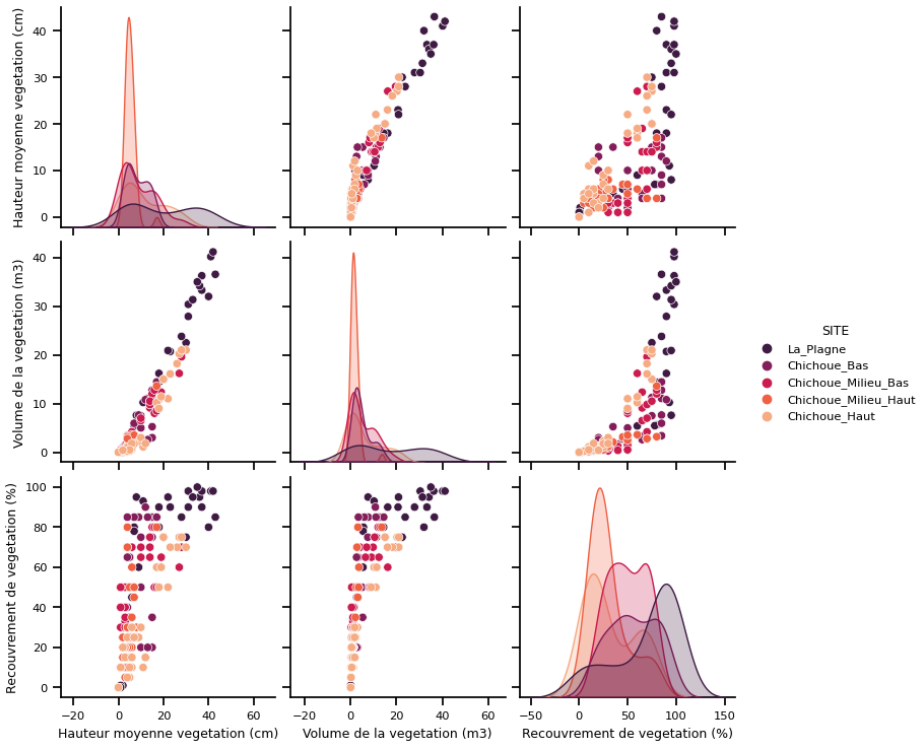

Figure 3: Pairwise relationships between variables

3 Methods

The following three approaches are evaluated systematically in the current work. They are ordered according to increasing model complexity and corresponding requirements regarding data amounts and computing resources.

3.1 Model A: Manual feature engineering & random forest regression

Given the low amount of data points, one way to solve the regression problem is to rely on less complex machine learning models instead of deep learning based computer vision. In remote sensing studies, the Random Forest (RF) classifier [17] proved to be a robust algorithm which has been used many times across a wide range of applications [18].

To apply its regression-related equivalent – the RF regressor – first a set of meaningful features has to be engineered. A simple automated tile cover classification distinguishing between vegetation, bare ground and shadowed areas is introduced first to allow focusing on the vegetated parts of the tiles. To this aim, image segments with mean NDVI values above a certain threshold are classified as vegetation. The classification of shadowed areas is based on consistently low reflectance values across the channels R, G, B and is necessary as the NDVI is distorted in these areas. Given the classified tiles, engineering two types of features seems to be reasonable in order to approximate the human vision process.

First, spectral features and their derivates, i.e. the share of vegetation, bare ground and shadow, may be used for the regression of vegetation parameters. The shares are calculated for three different NDVI thresholds (0.3, 0.4, 0.5) to let the RF algorithm choose the most informative variable.

Second, the spatial arrangement of these classes may be helpful as well. Especially for the variables vegetation height and volume, it seems plausible to predict different values for spatial clustering or fragmentation of vegetation. Different proxies may be used to characterise this spatial aspect. Among others, local and global correlation measures as well as textural metrics (e.g. Haralick, GLCM) are possible. For this study, the size and shape properties of objects derived from applying a felzenszwalb segmentation are used. Specifically, the mean value and standard deviation of the size and shape index of all objects classified as vegetation are taken into account as features.

In order to represent multi-scale information, the features are calculated for two segmentation scale levels (20, 200). In total, this makes a set of less than 50 features derived from the images fed into the RF regressor.

3.2 Model B: Autoencoder followed by small CNN

As a second approach, one could automate the feature extraction process by using an autoencoder (AE) on the unlabelled dataset. AEs are neural networks consisting of an encoder and decoder part aiming at reconstructing the input image based on an intermediate latent (i.e. encoded) representation [19,20]. By creating latent representations smaller than the input image size, the AE is forced to learn image features that are meaningful. Building on the encoding part, a small follow-up CNN performing the actual regression task can be constructed.

For the given task, simple AE architectures without any enhancements – e.g. denoising, sparse, variational AEs [21,22] – were trained in an end-to-end, non-layerwise fashion. The encoding part of all tested AE models consists of 3 convolutional layers and the AE latent representation was fixed to 32 x 42 x 42, which is about 1/10 of the original data size (4 x 333 x 333). This dimensionality of the latent feature space allows to create a simple regression CNN with less than 10 K parameters in the second step. The regression CNN starts with a 1D convolutional layer intended to reduce the number of feature maps to the ones needed for the task at hand. Subsequently, 3 2D convolutional layers are used again. Average pooling is applied to the output, the features are flattened, fed into a linear layer and sigmoid is applied to force the output to take values between 0 and 1. Rescaling by multiplying with factor 100 represents the final step to get the common value range of the variables of interest.

Altogether, model approach B represents the combination of standard elements to prevent overfitting in deep learning approaches. Using AEs in a first pre-task step to reduce the data dimensionality represents a well-established procedure. Batch normalisation [23], known for its regularising effect, is applied within the convolutional blocks for both the AE and CNN regression net. Together with testing of dropout [24] for the CNN part and the application of early stopping with patience of 10 or 25 epochs, a variety of elements commonly assessed as increasing generalisability are combined in model B.

3.3 Model C: SOTA-CNN based on data synthesis

An alternative to reducing the model complexity is an extensive expansion of the data basis. Applying simple standard architectures for image classification problems such as resnets usually requires at least a few thousand labelled images as input. Such an expansion of the database by up to two orders of magnitude can hardly be achieved using standard data augmentation procedures. However, given the current setting with a great number of unlabelled tiles, one can rely on a procedure of generating pseudo-labels for these tiles. This data augmentation process will be referred to as data synthesis in the following in order to demarcate it clearly from the conceptually different standard data augmentation procedures.

The attribution of pseudo-values is done in the following based on image similarity analyses. Using the resnet-18 pretrained on the ImageNet cutting of the last layer, 512 features are calculated for each UAV tile. The vicinity of two vectors in this feature space is assumed to indicate the structural similarity of images [25] so that unlabelled tiles with their vectors being close to vectors of labelled tiles can be given their values. However, to circumvent the issues arising from the high dimensionality of the input space and also to visualise and thereby validate results, vectors are mapped to 2D space first before performing this kind of nearest neighbour assignment of values.

t-Distributed Stochastic Neighbour Embedding (T-SNE) is used as an underlying dimensionality reduction technique [26]. Applying t-SNE to a feature set derived from images to assess similarities has already been performed by Castillo-Navarro et al. [27], for example. Compared to Principal Component Analysis (PCA) as a linear variance maximation technique, t-SNE preserves small pairwise similarities between inputs. The underlying technique relies on a gradient descent-based minimisation of a cost function between similarity measures in high- and low-dimensional space. Specifically, distances between points are mapped to probability functions and the Kullback-Leibler (KL) divergences between high and low dimensional points are reduced by optimising the locations of low dimensional points. This non-linear mapping with consideration of local similarities tends to result in better visualisations of data patterns in the low dimensional space. Furthermore, one can control the variance of the probability function applied in the high-dimensional space by the so-called perplexity to give more weight to local or global data patterns. The flexibility of the t-SNE comes with the drawback of non-deterministic mappings, a lack of applicability to new unseen data, difficulties in interpreting the low-dimensional mapping [28] and poor computational efficiency and scaling compared to techniques such as PCA.

The idea of pseudo-value assignment continues with applying the t-SNE algorithm to a small sub-portion of all tiles containing an equal number of labelled training tiles and unlabelled tiles. The mean nearest neighbour (NN) distance between all low dimensional points is calculated and used as a distance threshold for assigning values to unlabelled tiles. If the distance between an unlabelled point and one or multiple labelled ones is lower than the mean NN distance, a pseudo-value is calculated for the unlabelled one using the weighted average of the values of all the labelled tiles being considered. Leaving points, whose NN distance exceeds the mean NN, unlabelled aims at assigning pseudo-values only to those tiles that are highly likely to be similar to the labelled ones.

This procedure is repeated for all unlabelled tiles by repeatedly forming a subset of the data from the labelled and an equal number of unlabelled tiles and interpolating nearest neighbour values. To handle t-SNE inherent stochasticity, multiple trials are conducted for each data subset. The one with minimum KL divergence and maximum RMSE for the vegetation coverage

calculated between labelled point pairs less than the mean NN distance apart is chosen to be the best one used for pseudo-value assignment. Afterwards, the datasets are balanced ensuring a uniform quantile spacing for each individual variable using a random oversampling technique. Finally, the number of generated pseudo-values is capped at 5000 to limit the amount of data and to speed up the subsequent training of the SOTA CNN.

Considering the wide range of available SOTA CNNs [29,30], two comparatively simple architectures were chosen to be tested for performing the actual regression task. Relying on transfer learning, the pretrained versions of resnet-18 with 10 M parameters [31] and efficientnet-b0 with 5 M parameters [32] were taken. Both were modified by replacing the last layer with the regression block as already used for model B (flattening, linear layer, sigmoid, rescaling). To preserve the effect of pre-training during the initial training epochs, learning rates (LRs) for the pretrained cores of both models are multiplied by ¼.

4 Experimental settings

4.1 Train-Test split

It seems desirable to test both intra- and inter-site generalisability and transferability of trained models. For inter-site testing, all measurements and tiles for CMB were excluded from training and used for testing. As evident in Fig. xx, CMB is a site with a relatively high spectral and class representativity. Thus, by testing the results using CMB, the transferability of the results to sites similar in terms of vegetation parameters can be checked. The transferability to areas with completely different vegetation, however, cannot be assessed.

For intra-site testing, tiles were split into train, validation and test portions in a spatially non-overlapping manner. Whereas the labelled tiles are barely overlapping anyway, considering overlap is important in case of the spatially contiguous unlabelled tiles in order to avoid bias in the evaluation of the results. For the unlabelled tiles, a random walk based procedure was used to derive at train-val-test splits where each portion consists of several clusters of neighbouring tiles for each site. With this way of assigning tiles to the different portions, only about 20 % of all tiles have to be removed due to spatial overlap while a proper spatial distribution of tiles is guaranteed as well. The split ratio (train-val-test) was set to 70% - 15% - 15% for the unlabelled tiles for model B. For model C, the split-ratio is 99.2% - 0.4% - 0.4% due to the great amount of generated pseudo-labels for training whereas for validation and testing still only the in-situ-measurements are used. The split ratio for the labelled tiles for model A and B was set to be 60% - 20% - 20%. The relatively high percentage of validation and test tiles for the labelled tiles is due to the scarcity of data in order to have at least 20 samples for each portion. A randomised stratified sampling design guaranteeing the split proportions for each site was used to obtain a single train-val-test split. For the regression part these splits were made 10 times in order to perform cross-validation of the results. Due to the considerable inter-site variability (Fig. 2), this procedure is considered sufficient to achieve substantial representativeness of the individual portions with respect to the value distribution of the variables of interest without having to manually perform and/or supervise the process of train-val-test division. For the AE part the high number of tiles across train-val-test portions does not require any cross-validation.

4.2 Hyperparameter search

All subsequent tests on neural network related hyperparameters are evaluated using the mean absolute error (MAE) accuracy on the validation sets. To reduce the time spent on hyperparameter search, most evaluations have only been made for the variable vegetation coverage assuming in a simplifying manner that hyperparameters found for this variable are also suitable for the regression of the other variables. Only for model B and hyperparameters dropout and augmentation (4.2.1) large scale tests across all variables have been carried out.

4.2.1 Model B

Both parts of model B – the AE pre-task and the subsequent CNN regression net – require the specifications of some hyperparameters. For the AE, different learning rates (LRs) and optimisers are tested as hyperparameters first. 0.001, 0.01 and 0.1 are tested as potentially suitable LRs. For optimisers, Adam and Adam with weight decay (AdamW) are assessed. The test is run 10 times for each combination of LR & optimiser on a 10 % subset of the total dataset of unlabelled tiles. In a second step, two net architectures are compared based on the best LR-optimiser-combination. A design with the convolutional block consisting of a single convolutional layer is tested against one where a U-Net typical double convolution is used. As this second hyperparameter test is less extensive it is carried out on the total dataset directly.

For the CNN regression net, a two-step adjustment of hyperparameters is carried out as well. First, a total of eight different combinations out of LRs (0.01 & 0.1), optimisers (Adam, AdamW) and losses (L1, L2) is tested each 10-times on two different train-val-test splits to determine a suitable setting. Afterwards the best combination is applied to perform a complete assessment of four different models on all of the 10 train-val-test splits 5-times each. The four different models are created based on two different dropout rates for the conv layers (0, 0.2) and applying standard data augmentation techniques (no, yes). The data augmentation techniques which are applied for two of the four models are brightness and hue transforms. Considering the missing calibration of reflectance values across sites, it seems reasonable to aim at a model that is insensitive to small brightness and hue changes. Again, this second hyperparameter test with dropout and augmentation is assessed directly on all splits due to the low data amount and correspondingly fast training of a single model. Different than LR, optimiser and loss settings, which are standard hyperparameters – probably not needing any specific adjustments in the current case of a simple net architecture anyway – dropout and augmentation are assumed to be particularly relevant for the problem at hand. Therefore, its these two which are tested more extensively.

4.2.2 Model C

Prior to dealing with the hyperparameters of the net, model C requires to choose a suitable perplexity value for t-SNE. Therefore, an experiment on different perplexities ranging from 1 to 100 is conducted using only a single sub data set with all labelled tiles and a same amount of randomly selected unlabelled tiles as input. t-SNE is performed 100 times for every perplexity value. Assessing the corresponding t-SNE embeddings is done using the average RMSE value for vegetation coverage measured across all labelled point pairs with separating distances smaller than the mean NN distance. Additionally, the percentage of unlabelled tiles that would be labelled following the threshold-controlled nearest neighbour procedure as explained in 3.3 is taken into account.

Testing the two net architectures resnet-18 and efficientnet-b0 was done in one go together with the adjustment of the hyperparameters LRs (0.01, 0.001), optimisers (Adam, AdamW) and losses (L1, L2), For each combination of hyperparameters 5 tests were carried out, each of them on different train-val-test splits.

4.3 Evaluation metrics & settings

For the regression task, the following metrics are used to assess the accuracy of results: mean absolute error (MAE), mean squared error (MSE), Person's correlation coefficient (r), coefficient of determination (R2) and Spearman's correlation coefficient (s). For the evaluation of AE-based reconstruction abilities, five different metrics are used: MAE, MSE, angle between spectra (SAM), peak signal noise ratio (PSNR) and structural similarity index measure (SSIM) [1]. Including the kernel-based SSIM aims at providing not only information on absolute spectral accuracies but also the reconstruction accuracy of spatial dependencies.

Trainings of the final models with determined hyperparameters as well as their corresponding evaluations on the test data set were generally performed multiple times (exception: model B, AE part 5.1.1/5.2.2). This aims at assessing the stability and reproducibility of the trainings and enabling the selection of the best performing model based on the validation metrics to avoid unfavourable local minima solutions.

[1] SSMI was not calculated in all experiments. As the calculation is computationally heavier, only trainings on the LETG server include this metrics, not those performed on my personal computer.

5 Results

5.1 Hyperparameter definition

5.1.1 Model B

For the hyperparameter tests of the AE part, 4 out of the top 10 models with lowest MAE use AdamW with 0.01 learning rate and require around 100 epochs until convergence. 3 out of the top 10 models rely on Adam with 0.01 and on average need a few more than 100 epochs until convergence. Thus, a LR of 0.01 can be identified as preferable. Also, AdamW seems to achieve slightly faster & potentially better results with the drawback of a slightly increased variability across runs. Still, the combination of LR 0.01 and AdamW is set as hyperparameter combination for all further experiments. Comparing the results of single and double conv layers, one consistently obtains lower MAEs for the single conv layer setting (Table 2). It is also evident that for each setting there is low variation in the results across different trials such that multiple runs of the final model are not even necessary. Instead, a single run with the final hyperparameter constellation on the total data set seems to be sufficient.

Table 2: Results of hyperparameter tests for AEs

model run	MAE single conv	MAE double conv
1	0.0242	0.0319
2	0.0242	0.0304
3	0.0240	0.0314

For the regression CNN, on the other hand, results do not only vary between different hyperparameter settings but also across trials for the same setting. This is expressed in partly considerable margins between 25%-quantile and median MAE values for a given setting in Table 3. This can be attributed to the fact that experiments are done on different train-val-test splits other than for the AE test, where the same data set was used consistently. Still, even for the same train-val-test splits, results differ significantly in many cases which is a consequence of the overall small data size compared to model complexity. This matches the observation of fluctuating validation loss curves. Even if this complicates the evaluation of hyperparameter tests, Table. 3 indicates that the setting with LR 0.001, Adam as optimiser and L1 loss achieves good results in terms of both 25%-quantile and median MAE.

Table 4 presents an analogous summary of the results for the subsequent tests dealing with the influences of augmentation and dropout. Apparently, models without augmentation and dropout already produce good results across different trials and test-val-splits. Especially for the variables height and volume, MAEs are increasing when including augmentation and/or dropout. Only for vegetation coverage validation accuracies on the augmented dataset tend to produce slightly better results than on non-augmented ones. Still, due to the other variables the most chosen setting for the final evaluation on the test set is set to no augmentation and no dropout.

Table 3: Results of hyperparameter tests for CNN regression nets (part I).
Accuracy figures are calculated across 20 trials per setting.

lr	optimiser	loss	MAE 25%-quantile	MAE median
0.001	Adam	L1	8.2	8.9
0.001	AdamW	L2	8.4	9.3
0.01	Adam	L1	8.1	9.4
0.01	Adam	L2	8.8	9.8
0.001	Adam	L2	8.2	9.8
0.001	AdamW	L1	7.9	10.1
0.01	AdamW	L1	8.0	10.2
0.01	AdamW	L2	9.1	12.1

Table 4: Results of hyperparameter tests for CNN regression nets (part II).
Accuracy figures are calculated across 50 trial per setting.

variable	augment	dropout	MAE 25%-quantile	MAE median
coverage	False	0	8.6	9.5
		0.2	9.0	9.8
	True	0	8.3	9.5
		0.2	8.8	9.9
height	False	0	4.3	5.2
		0.2	5.3	6.1
	True	0	5.2	6.3
		0.2	5.4	6.6
volume	False	0	3.3	4.0
		0.2	4.5	5.5
	True	0	4.2	5.2
		0.2	5.0	5.8

Fig. 4 allows a more detailed assessment of the results of the second hyperparameter test also in terms of individual model runs. For all the variables and hyperparameter settings a certain range of MAEs is evident. This is expected and partly to due the evaluation on different train-val-test splits. For very few models large MAEs are visible. The origin of these deviating model runs remains to be investigated and may be attributed to some unfavourable train-val-test splits. The fact that this concerns less than 15 out of a total of 600 models, however, won't affect the models performance in the proposed training design with multiple repetitions and confirms the general suitability of the chosen hyperparameter settings.

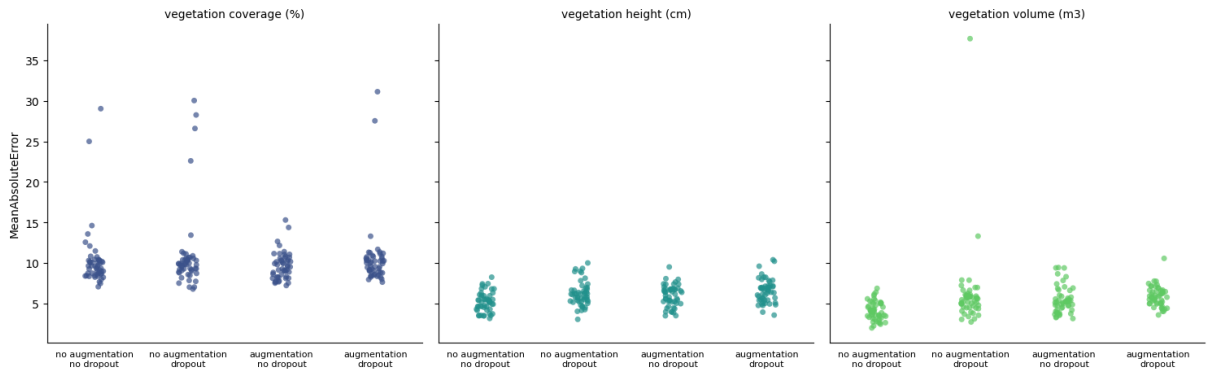

Figure 4: MAE for all models (model type B) per hyperparameter constellation and variable of interest

Putting encoder and CNN regression part together considering all the above-mentioned hyperparameters affecting the model architecture, one derives at a regression net with about 21 K parameters for model B.

Table 5: Final architecture model B

Layer (type:depth-idx)	Input Shape	Output Shape	Param #	Kernel Shape
├─Sequential: 1-1	[-1, 4, 333, 333]	[-1, 32, 42, 42]	--	--
│ └─Sequential: 2-1	[-1, 4, 333, 333]	[-1, 16, 167, 167]	--	--
│ │ └─Conv2d: 3-1	[-1, 4, 333, 333]	[-1, 16, 167, 167]	(576)	[4, 16, 3, 3]
│ │ └─BatchNorm2d: 3-2	[-1, 16, 167, 167]	[-1, 16, 167, 167]	(32)	[16]
│ │ └─ReLU: 3-3	[-1, 16, 167, 167]	[-1, 16, 167, 167]	--	--
│ └─Sequential: 2-2	[-1, 16, 167, 167]	[-1, 32, 84, 84]	--	--
│ │ └─Conv2d: 3-4	[-1, 16, 167, 167]	[-1, 32, 84, 84]	(4,608)	[16, 32, 3, 3]
│ │ └─BatchNorm2d: 3-5	[-1, 32, 84, 84]	[-1, 32, 84, 84]	(64)	[32]
│ │ └─ReLU: 3-6	[-1, 32, 84, 84]	[-1, 32, 84, 84]	--	--
│ └─Sequential: 2-3	[-1, 32, 84, 84]	[-1, 32, 42, 42]	--	--
│ │ └─Conv2d: 3-7	[-1, 32, 84, 84]	[-1, 32, 42, 42]	(9,216)	[32, 32, 3, 3]
│ │ └─BatchNorm2d: 3-8	[-1, 32, 42, 42]	[-1, 32, 42, 42]	(64)	[32]
│ │ └─ReLU: 3-9	[-1, 32, 42, 42]	[-1, 32, 42, 42]	--	--
├─regression_cnn: 1-2	[-1, 32, 42, 42]	[-1, 1]	--	--
│ └─Sequential: 2-4	[-1, 32, 42, 42]	[-1, 32, 6, 6]	--	--
│ │ └─Sequential: 3-10	[-1, 32, 42, 42]	[-1, 4, 44, 44]	136	--
│ │ └─Sequential: 3-11	[-1, 4, 44, 44]	[-1, 8, 22, 22]	304	--
│ │ └─Sequential: 3-12	[-1, 8, 22, 22]	[-1, 16, 11, 11]	1,184	--
│ │ └─Sequential: 3-13	[-1, 16, 11, 11]	[-1, 32, 6, 6]	4,672	--
│ └─Sequential: 2-5	[-1, 32, 6, 6]	[-1, 1]	--	--
│ │ └─AvgPool2d: 3-14	[-1, 32, 6, 6]	[-1, 32, 1, 1]	--	--
│ │ └─Flatten: 3-15	[-1, 32, 1, 1]	[-1, 32]	--	--
│ │ └─Linear: 3-16	[-1, 32]	[-1, 1]	33	[32, 1]
│ │ └─Sigmoid: 3-17	[-1, 1]	[-1, 1]	--	--

Total params: 20,889

5.1.2 Model C

Exemplary results of the t-SNE based 2D representations of labelled images are shown below for 5 different perplexities with each embedding calculated 5 times. Considering the local clustering of similar vegetation values as well as the existing global gradient across the 2D space, the use of deep features as perceptual similarity metrics seems to work in a reliable manner. Only for the smallest perplexity of 2, local similarities are weighted too heavy for representing global patterns but even in this case the local clustering of similar values would still allow the application of the proposed nearest neighbour based pseudo-value assignment.

Assessing the RMSE systematically based on 100 trials per perplexity value, the lowest mean RMSE values smaller than 12 are achieved for perplexities smaller 5 and perplexities between 20 and 35. For large ones (> 50) the mean RMSE rises above 13. Note that given the small amount of data (less than 200 data points), the results of the t-SNE are not changing substantially anymore once the perplexity is large enough (Fig. 5). Analysing the 10%-quantile of RMSE values largely agrees with the results of mean RMSE. The 10%-quantile RMSE drop below 10 for very small perplexity values and values in the range 20 to 30. Including the observation that the percentage of unlabelled tiles that could be labelled rises continuously from less than 20 % to almost 40 % with increasing perplexities, one can conclude that the medium perplexity values are preferable. For all subsequent experiments, the perplexity was therefore set to 25.

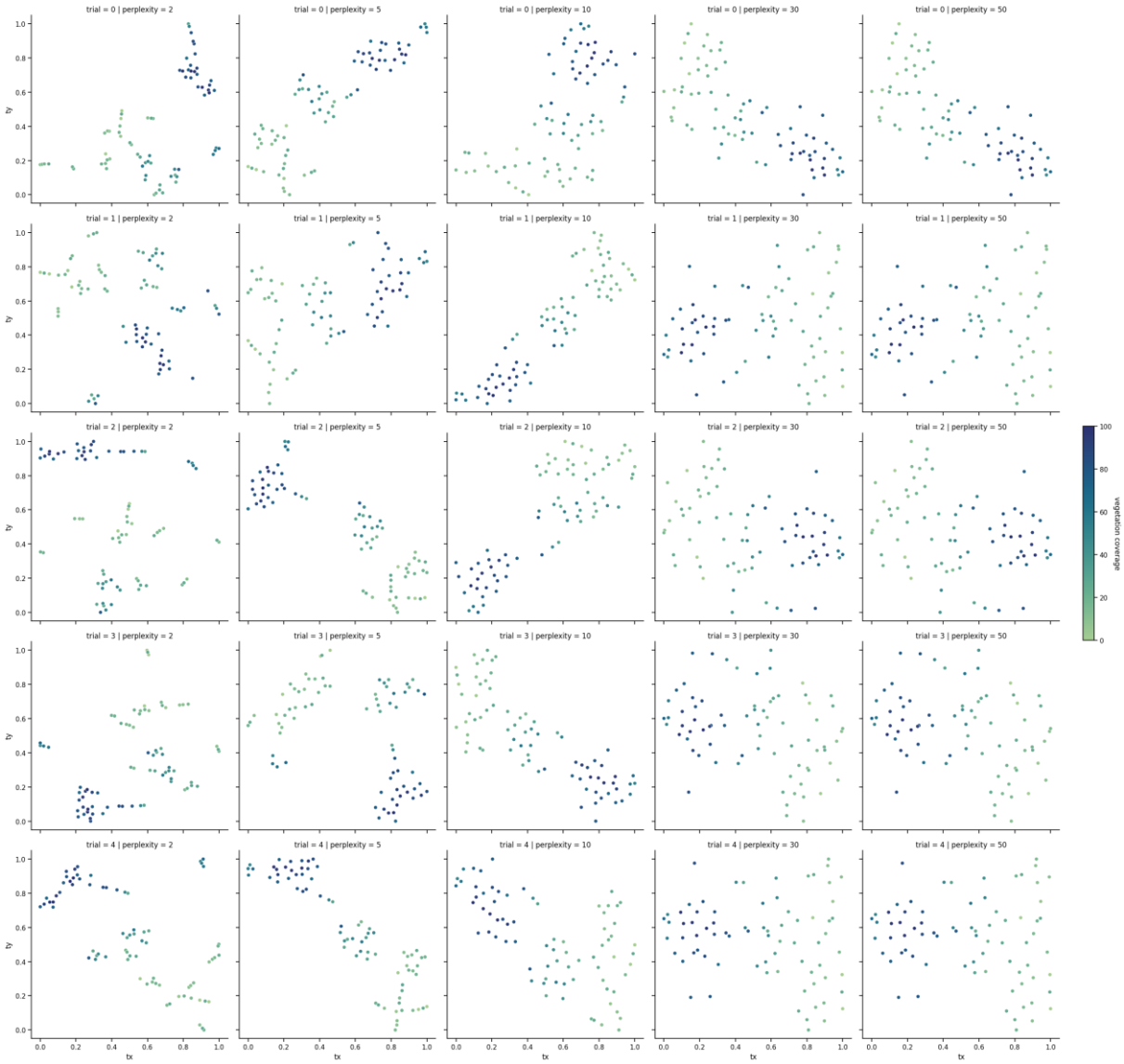

Figure 5: TSNE results for a range of perplexities & several repetitions

Table 6: Results of hyperparameter tests for SOTA nets.
Accuracies are calculated across the 5 trials per setting.

net	lr	optimiser	loss	MAE 25%-quantile	MAE median
efficientnet	0.001	AdamW	L1	5.3	5.9
efficientnet	0.01	AdamW	L1	5.9	6.2
resnet	0.001	AdamW	L1	6.0	6.3
efficientnet	0.001	Adam	L2	6.2	6.4
efficientnet	0.001	AdamW	L2	6.2	6.5
resnet	0.001	Adam	L2	6.5	6.6
efficientnet	0.01	Adam	L1	6.2	6.6
efficientnet	0.01	Adam	L2	6.0	6.7
efficientnet	0.001	Adam	L1	5.2	6.7
resnet	0.001	AdamW	L2	6.4	6.7
efficientnet	0.01	AdamW	L2	6.5	6.8
resnet	0.001	Adam	L1	5.8	7.2
resnet	0.01	AdamW	L2	7.3	7.3
resnet	0.01	Adam	L2	7.6	7.7
resnet	0.01	Adam	L1	6.8	7.9
resnet	0.01	AdamW	L1	7.5	8.1

Assessing hyperparameter constellations for the regression nets (Table 6), it is evident that LRs of 0.01 consistently lead to lower errors on the validation set for various constellations of the other hyperparameters. Regarding optimisers & loss functions no clear patterns regarding superior settings are visible. With respect to net architectures, efficientnets tend to perform better than resnets. However, due to the computational efficiency [2] and an acceptable sacrifice of accuracy, resnet is chosen as a net architecture for the further evaluation of model C. The other hyperparameters are set according to the best-performing resnet (Tab. 1, 3rd row).

5.2 Performance of models

An overview on the performance of all models is shown below. The different matrices essentially agree with each other in terms of their informative value about model quality. Vegetation coverage is predicted most reliable by all models with mean R2 values consistently above 0.85. For the other variables lower shares of explained variance in the range of 0.5 – 0.6 (vegetation volume) and 0.3 – 0.5 (vegetation heights), respectively, are obtained. Comparisons across the different models indicate the superiority of the data synthesis based approach (model C) in all cases. Its mean MAE is slightly lower than those achieved in case of the random forest (model A) or the AE-based approach (model B). Complementary, higher correlation coefficients are measured as well. Model A and B tend to perform similar for vegetation volume and heights whereas A performs better compared to B for the vegetation coverage regression. However, differences between models are rarely significant. Given the small number of test samples and the limited number of folds for cross-validation (10 train-val-test splits), the 95 % confidence intervals around the mean metrics are relatively large. A detailed view on the speed and accuracy performance of each model is presented subsequently.

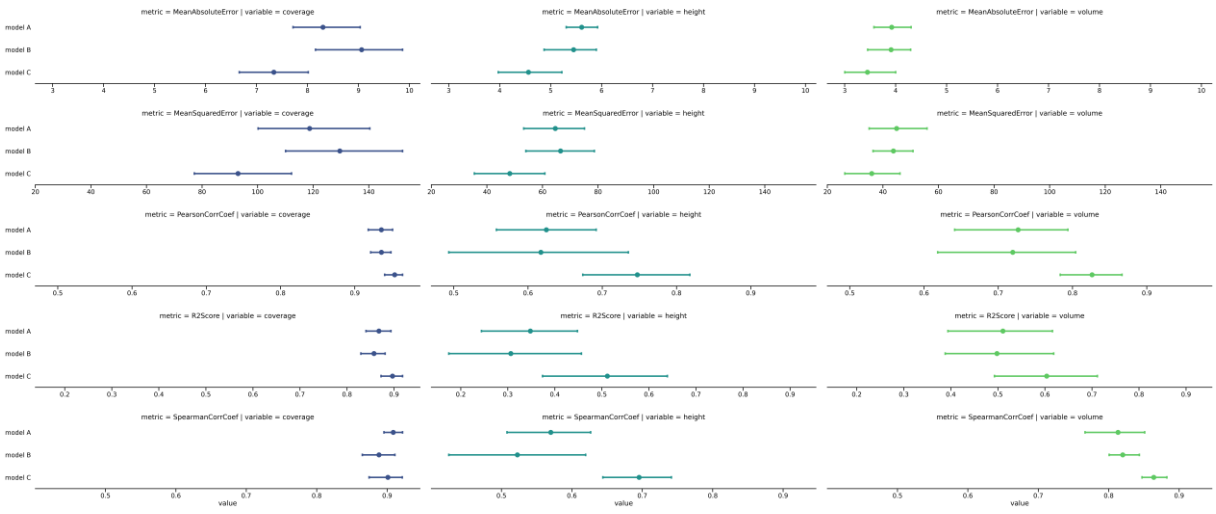

Figure 6: Accuracies for all models & variables. Means across the 10 train-val-test splits together with 95 % confidence intervals (CI) are displayed.

[2] Even though efficientnet-b0 has less parameters, its pytorch implementation takes about 2-3 times compared to training resnet-18, presumably as depth-wise convolution is pretty slow, for further discussion see here

5.2.1 Model A

Applying object-based multi-scale feature engineering for a single tile takes a few seconds. Conducting the random forest regression multiple times for all train-test-val splits takes less than 10 min. Therefore, the execution of the complete feature engineering – training – evaluation pipeline needs approximately ½ hour for the given number of tiles (i.e. n = 160). Intermediate results of the feature engineering process are shown in Fig. 7 exemplarily for a single tile.

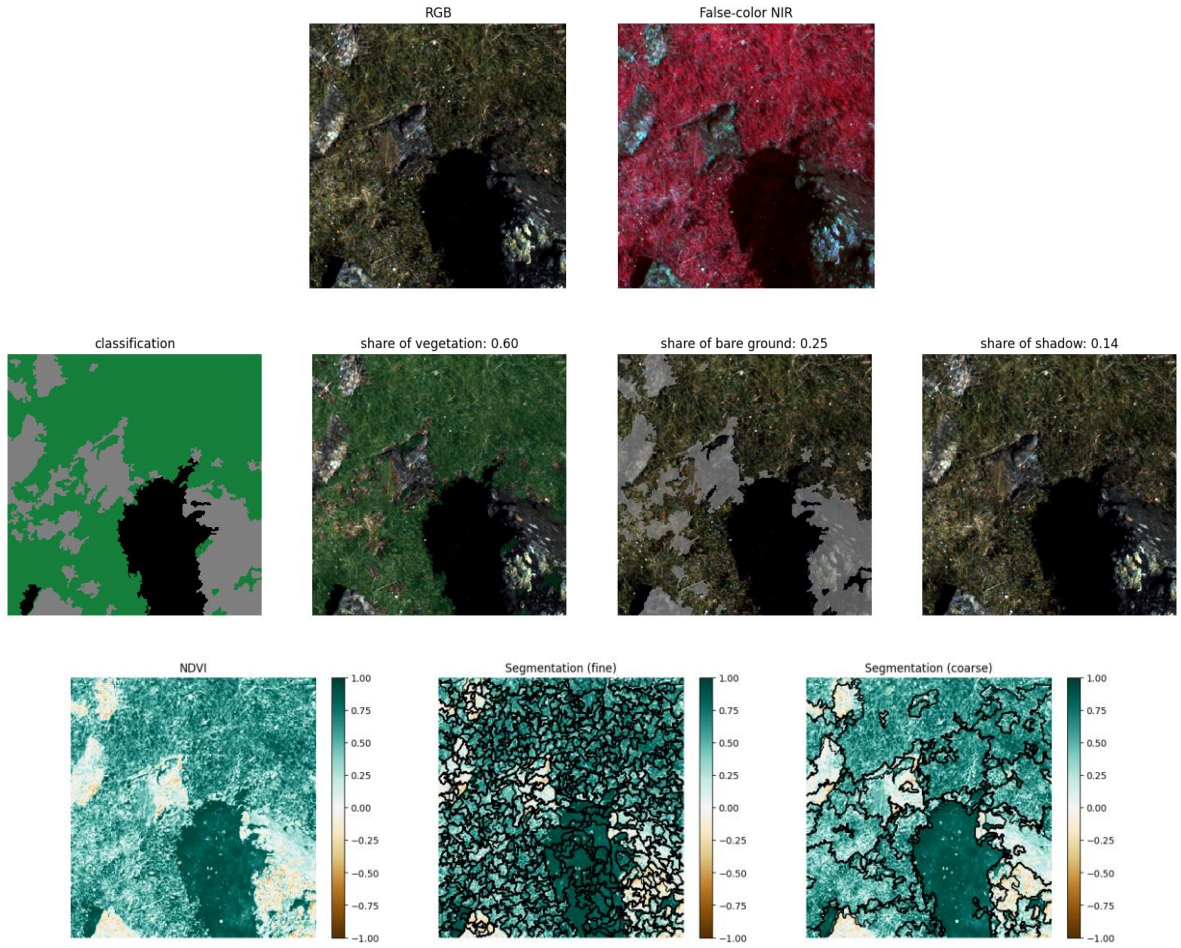

Figure 7: Exemplary intermediate products during the feature engineering process model A. Original RGB visualisation and false-color NIR, R, G are shown on the top. In the middle, the output of the classification based on a NDVI threshold of 0.4 is displayed. The final row shows the two levels of segmentations relying on scale parameters 20 and 200.

Table 7 shows the mean accuracies achieved on the tests set as displayed also in Fig. 6. A further breakdown of the results for individual train-test-val splits using the MAE metric is shown in Fig. 8.

Table 7: Performance results for model A. Mean values across all train-test-val splits are displayed

variable	MAE	MSE	r	s	R2
coverage	8.3	118.7	0.94	0.91	0.87
height	5.6	64.6	0.62	0.57	0.35
volume	3.9	44.9	0.73	0.81	0.51

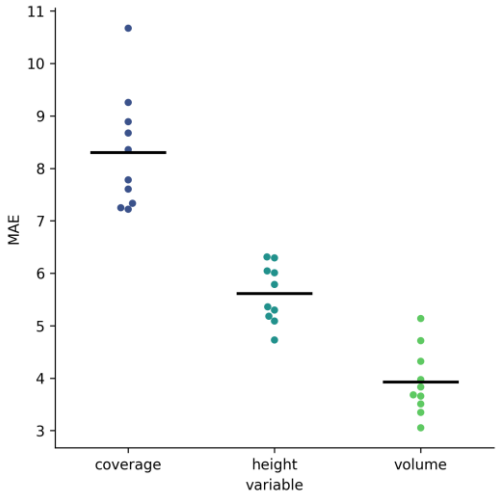

Figure 8: MAE achieved for individual train-val-test splits (model A). The black line indicates the average.

5.2.2 Model B

For the AE part, training on the total set of 18.000 images and 4000 tiles each for validation and testing takes about 2.5 min/epoch [3]. Training and validation loss curves for a single training with the hyperparameters according to 5.1.1 show a relatively smooth decay (Fig. 9). Convergence is reached after only a few epochs.

Accuracy figures for the test set are shown in Table 8. Considering the value range of [0,1], a MAE of 0.025 indicates a good approximation of original images by reconstructed tiles. This view is supported by the other metrics as well as a visual assessment of the results (Fig. 9).

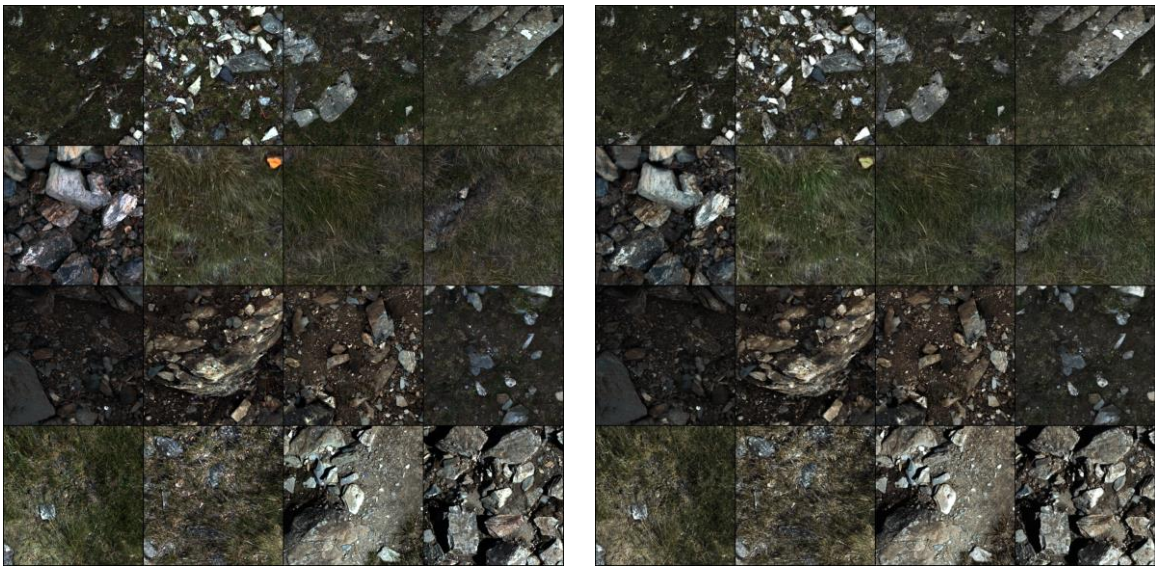

Figure 9: Comparison of original input and reconstructed images for a set of 16 tiles belonging to the test set with 4 tiles randomly selected for each site

[3] This is the timing achieved on my personal computer (GPU: NVIDIA GeForce RTX 3060 Laptop GPU GPU memory: 6.0 GiB), not directly comparable to all the other experiments conducted on the larger server.

Table 8: Performance results for model B (AE part)

metric	accuracy
MAE	0.025
MSE	0.001
SAM	0.083
PSNR	28.97

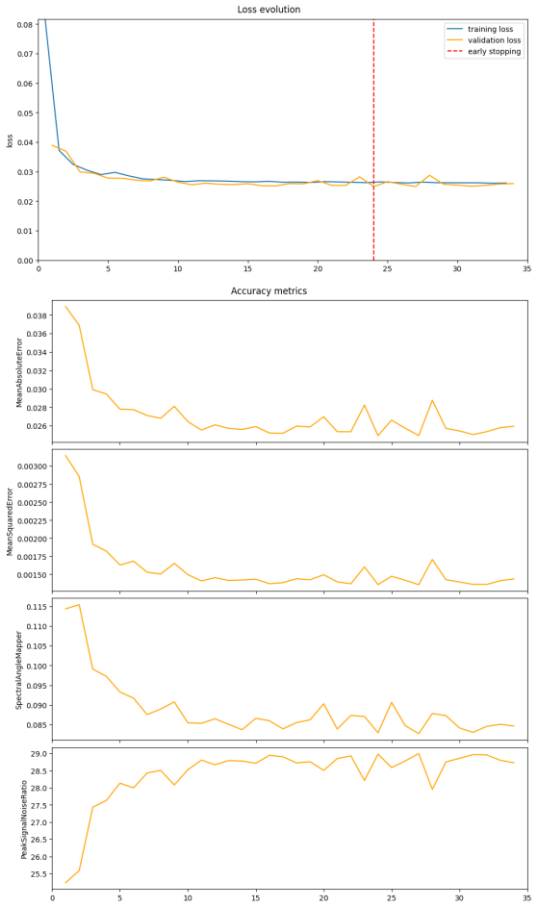

Figure 10: Loss and accuracy development during training (model B, AE part)

Training the small regression net stacked on top of the CNN on the small set of 160 tiles is rather fast with a few seconds per epoch. Training and validation are characterised by an early decrease in training loss and a delayed decrease in validation loss (Fig. 11). Convergence is reached after 50 – 100 epochs in many cases. The following results on the variables of interest are achieved in terms of accuracies on test sets (see also Fig. 6).

Table 9: Performance results for model B (CNN-based regression). Mean values across all train-test-val splits are displayed.

variable	MAE	MSE	r	s	R2
coverage	9.1	129.6	0.94	0.89	0.86
height	5.5	66.5	0.62	0.52	0.31
volume	3.9	43.8	0.72	0.82	0.50

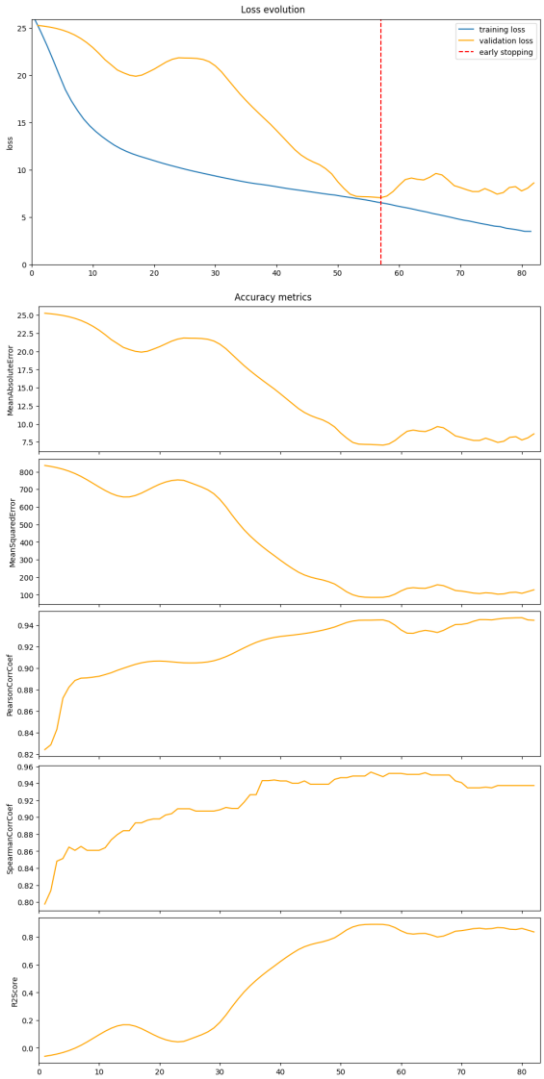

Figure 11: Loss and accuracy development during training (model B, CNN-based regression). A representative example was chosen (best model for split nr 10, variable coverage).

5.2.3 Model C

Training on the total set of 5.000 tiles takes about 0.5 min/epoch. Most trainings are finished in less than 30 epochs. Inspecting the loss curves reveals that validation accuracies move rather erratically whereas the training loss is decreasing continuously in a smooth manner (Fig. 12). Table 10. shows the mean accuracies achieved on the tests set – again, these numbers are visualised in Fig. 6 as well.

Table 10: Performance results for model C. Mean values across all train-test-val splits are displayed.

variable	MAE	MSE	r	s	R2
coverage	7.3	92.9	0.95	0.90	0.90
height	4.6	48.2	0.75	0.70	0.51
volume	3.4	36.0	0.83	0.86	0.60

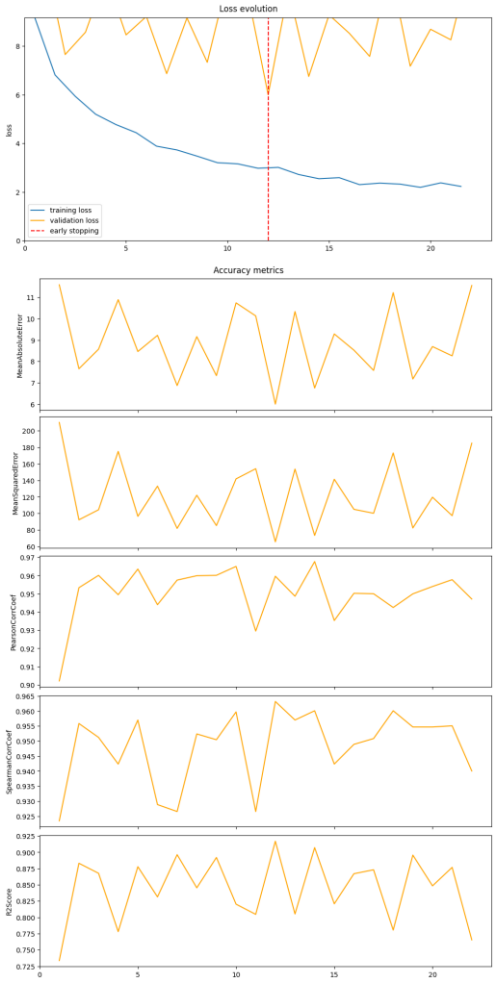

Figure 12: Loss and accuracy development during training (model C). A representative example was chosen (best model for split nr 10, variable coverage).

6 Discussion & Conclusion

An extended discussion of the results is still pending. For the time being, however, the following points can be summarised:

Overall model C seems to perform slightly better than the other models. The benefit of model C can be seen in the possibility to utilise the full capacity of a SOTA-CNN without reducing the information and model complexity first. Data synthesis is significantly more powerful than common data augmentation techniques as the data set is enhanced by morphologically similar but still new tiles. Moreover, this approach is the only one that allows to balance the dataset in the given sparse training data setting. Even though, the influence of balancing the data set regarding model performance has not been assessed systematically in the current work, a higher degree of generalisability appears evident.

At the same time, model C is the model with the most obvious potential for improvements and thus easily achievable further performance gains. As explained earlier, to limit the computational effort, data synthesis was capped at 5 K generated samples. However, the given extent of the UAV tiles allows to generate up to 20 K samples expected to result in further improvements of the model. Also, the current evaluation settings on the validation are disadvantageous leading to sub-optimal model choices. The comparatively large size of the training set relative to the validation set manifests itself in the presented fluctuating loss curve in case the accuracy assessment on the validation set takes place after a full run over all training batches. Instead, an evaluation after a few batches seems reasonable to ensure a smooth loss curve as a prerequisite for finding a good solution via early stopping. Currently, due to missing this, the correlations between validation and test metrics are quite low meaning that the selection of the final model configuration as the one performing best on the validation set is hardly better than selecting any trained model configuration.

From a user's point of view, it has to be decided whether the performance gain of model C relative to the other models justifies the associated processing effort with corresponding time and resource input. In the sense of this holistic perspective, Table xx gives a general overview of relevant comparison criteria beyond the accuracy orientation. The machine learning based model A, for example, is characterised by the fact that its setup requires very little effort and still achieves good results. Only the prediction effort at runtime is comparatively high due to the need to calculate object-oriented features for each tile of interest. However, considerable speed improvements may be realised easily by implementing multi-core solutions. Based on the analyses conducted so far, the autoencoder-based approach (model B) has not proven successful, as the model does not outperform the model A despite considerable additional development effort.

Table 11: Summarising comparison of the models

		model A	model B	model C
development perspective	training setup effort *(implementation effort, data preparation, hyperparameter adjustment)*	low	high	high
↓	training speed & training resources *(processing power)*	fast/low	medium	medium
↓	prediction effort *(speed of use of the trained model)*	high	medium	medium
application perspective	accuracy *(performance in terms of metrics)*	medium	medium	high

7 Outlook & Future efforts

In the following, additional steps considered significant are prioritised, also in relation to a possible publication of this work:

The following things need to be done to complete previous analyses (*must*):

- Solid literature review with focus on approaches for regression of variables of interest & identification of foci so far, e. g. vegetation coverage via regression with NDVI, vegetation height via surface models derived from point clouds, etc
- Extended background information on data acquisition & processing
    - Timing of data acquisitions, drones used, etc
    - Process of orthophoto generation
- Systematic hyperparameter tests for Model B, esp. the AE part, presented results above have been obtained by using a first version of the dataset with 25% overlap instead of 50% overlap during tiling, also loss was not tested – set of tested hyperparameters should be consistent across all the models
- Better systematics for hyperparameter tests for regression parts (evaluation metrics median and 25%-quantiles suitable?, tests across different train-test-val splits each with multiple trainings per specific split reasonable?)
- Inclusion of learning rate weights between pre-trained net and following regression block as crucial hyperparameter into corresponding tests for model C (¼ chosen currently not based on systematic tests)
- Retraining model C with higher frequency of validation set evaluation after a subset of the training data to ensure a smooth loss curve
- Inter-site testing for CMB & generation of continuous map of predicted values for this site
- Scatterplot of predicted vs. in-situ measurements & visual evaluations of tiles with large differences between predicted & in-situ measurements
- In depth discussion of results – also for hyperparameter tests, e.g. unexpected result of reduced performance of model B in case of standard hue & brightness transform for data augmentation)

The subsequent things are worth taking a deeper look into (*should*):

- Inclusion of Bocard data once the data is reprocessed flawlessly
- Use of more sophisticated AE architectures: stacked, sparse, denoising, variational

- Further compression of information in latent representation of AE part, down to a dimensionality which may also be directly fed into a classical ML algorithm instead of a CNN regression net
- Inclusion of structural 3D information from UAV as input data, i.e. height information derived from further photogrammetric processing of UAC imagery

The subsequent things may be worth taking a deeper look into (*could*):

- Speeding up the pseudo-value assignment by relying on UMAP instead of t-SNE, so far an already efficiency-optimised multi-core implementation of t-SNE is used [33], however creating a single train-test-val split (with 5000 samples) still takes about ½ hours, apart from the computational burden considering UMAP may be interesting from a methodological point of view, too – as described in 5.1.1 local similarities are not necessarily important for the given task such that UMAP, tending to have a stronger focus on global data patterns, may be a viable alternative
- Model B: combining dropout-in/excluding models together with augmenting/non-augmenting models as an ensemble model
- Model C: weighted loss function with emphasis on labels over pseudo-labels for data synthesis approach

References

1. Maes, W.H.; Steppe, K. Perspectives for Remote Sensing with Unmanned Aerial Vehicles in Precision Agriculture. *Trends Plant Sci.* **2019**, *24*, 152–164, doi:10.1016/j.tplants.2018.11.007.
2. Wang, T.; Liu, Y.; Wang, M.; Fan, Q.; Tian, H.; Qiao, X.; Li, Y. Applications of UAS in Crop Biomass Monitoring: A Review. *Front. Plant Sci.* **2021**, *12*, 616689, doi:10.3389/fpls.2021.616689.
3. Zhao, C. Unmanned Aerial Vehicle Remote Sensing for Field-Based Crop Phenotyping: Current Status and Perspectives.
4. Gao, L.; Wang, X.; Johnson, B.A.; Tian, Q.; Wang, Y.; Verrelst, J.; Mu, X.; Gu, X. Remote sensing algorithms for estimation of fractional vegetation cover using pure vegetation index values: A review. *ISPRS Journal of Photogrammetry and Remote Sensing* **2020**, *159*, 364–377, doi:10.1016/j.isprsjprs.2019.11.018.
5. Purevdorj, T.; Tateishi, R.; Ishiyama, T.; Honda, Y. Relationships between percent vegetation cover and vegetation indices. *International Journal of Remote Sensing* **1998**, *19*, 3519–3535, doi:10.1080/014311698213795.
6. Daniela Stojanova; Panče Panov; Valentin Gjorgjioski; Andrej Kobler; Sašo Džeroski. Estimating vegetation height and canopy cover from remotely sensed data with machine learning.
7. G. Poley, L.; J. McDermid, G. A Systematic Review of the Factors Influencing the Estimation of Vegetation Aboveground Biomass Using Unmanned Aerial Systems. *Remote Sensing* **2020**, *12*, 1052, doi:10.3390/rs12071052.
8. Li, W.; Niu, Z.; Chen, H.; Li, D.; Wu, M.; Zhao, W. Remote estimation of canopy height and aboveground biomass of maize using high-resolution stereo images from a low-cost unmanned aerial vehicle system. *Ecological Indicators* **2016**, *67*, 637–648, doi:10.1016/j.ecolind.2016.03.036.
9. Rueda-Ayala, V.P.; Peña, J.M.; Höglind, M.; Bengochea-Guevara, J.M.; Andújar, D. Comparing UAV-Based Technologies and RGB-D Reconstruction Methods for Plant Height and Biomass Monitoring on Grass Ley. *Sensors (Basel)* **2019**, *19*, doi:10.3390/s19030535.
10. Viljanen, N.; Honkavaara, E.; Näsi, R.; Hakala, T.; Niemeläinen, O.; Kaivosoja, J. A Novel Machine Learning Method for Estimating Biomass of Grass Swards Using a Photogrammetric Canopy Height Model, Images and Vegetation Indices Captured by a Drone. *Agriculture* **2018**, *8*, 70, doi:10.3390/agriculture8050070.
11. Zhang, H.; Sun, Y.; Chang, L.; Qin, Y.; Chen, J.; Qin, Y.; Du, J.; Yi, S.; Wang, Y. Estimation of Grassland Canopy Height and Aboveground Biomass at the Quadrat Scale Using Unmanned Aerial Vehicle. *Remote Sensing* **2018**, *10*, 851, doi:10.3390/rs10060851.

12. Osco, L.P.; Marcato Junior, J.; Marques Ramos, A.P.; Castro Jorge, L.A. de; Fatholahi, S.N.; Andrade Silva, J. de; Matsubara, E.T.; Pistori, H.; Gonçalves, W.N.; Li, J. A review on deep learning in UAV remote sensing. *International Journal of Applied Earth Observation and Geoinformation* **2021**, *102*, 102456, doi:10.1016/j.jag.2021.102456.
13. Brigato, L.; Iocchi, L. A Close Look at Deep Learning with Small Data **2020**.
14. Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; Vinyals, O. Understanding deep learning requires rethinking generalization **2017**.
15. Kukačka, J.; Golkov, V.; Cremers, D. Regularization for Deep Learning: A Taxonomy **2017**.
16. Delerue, F. Positive Plant-Plant interactions and spatial Patterns in Pyrenean Post-mine tailings (SixP). Available online: https://sixp.inria.fr/en/ (accessed on 1 September 2022).
17. Breiman, L. Random Forests. *Machine Learning* **2001**, *45*, 5–32, doi:10.1023/A:1010933404324.
18. Belgiu, M.; Drăguţ, L. Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing* **2016**, *114*, 24–31, doi:10.1016/j.isprsjprs.2016.01.011.
19. Bank, D.; Koenigstein, N.; Giryes, R. *Autoencoders*, 2020. Available online: http://arxiv.org/pdf/2003.05991v2.
20. Wang, Y.; Yao, H.; Zhao, S. Auto-encoder based dimensionality reduction. *Neurocomputing* **2016**, *184*, 232–242, doi:10.1016/j.neucom.2015.08.104.
21. Dong, G.; Liao, G.; Liu, H.; Kuang, G. A Review of the Autoencoder and Its Variants: A Comparative Perspective from Target Recognition in Synthetic-Aperture Radar Images. *IEEE Geosci. Remote Sens. Mag.* **2018**, *6*, 44–68, doi:10.1109/MGRS.2018.2853555.
22. Tschannen, M.; Bachem, O.; Lucic, M. *Recent Advances in Autoencoder-Based Representation Learning*, 2018. Available online: http://arxiv.org/pdf/1812.05069v1.
23. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift **2015**.
24. Srivastava, N.; Hinton, G.E.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* **2014**, *15*, 1929–1958.
25. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*, 2018. Available online: http://arxiv.org/pdf/1801.03924v2.
26. van der Maaten, L.; Hinton, G.E. Visualizing Data using t-SNE. *Journal of Machine Learning Research* **2008**, 2579–2605.
27. Castillo-Navarro, J.; Le Saux, B.; Boulch, A.; Audebert, N.; Lefèvre, S. *Semi-Supervised Semantic Segmentation in Earth Observation: The MiniFrance Suite, Dataset Analysis and Multi-task Network Study*, 2020. Available online: http://arxiv.org/pdf/2010.07830v1.

28. Wattenberg, M.; Viégas, F.; Johnson, I. How to Use t-SNE Effectively. *Distill* **2016**, *1*, e2, doi:10.23915/distill.00002.
29. Khan, A.; Sohail, A.; Zahoora, U.; Qureshi, A.S. A survey of the recent architectures of deep convolutional neural networks. *Artif Intell Rev* **2020**, *53*, 5455–5516, doi:10.1007/s10462-020-09825-6.
30. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 53, doi:10.1186/s40537-021-00444-8.
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition **2015**.
32. Tan, M.; Le V, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *International Conference on Machine Learning* **2019**.
33. Ulyanov, D. Parallel t-SNE implementation with Python and Torch wrappers. Available online: https://github.com/DmitryUlyanov/Multicore-TSNE (accessed on 1 September 2022).