# GEOMETRY-BASED POINT CLOUD CLASSIFICATION USING HEIGHT DISTRIBUTIONS

F. Politz [1],*, M. Sester [1], C. Brenner [1]

[1] Institute of Cartography and Geoinformatics, Leibniz University Hannover, Germany - (florian.politz, monika.sester, claus.brenner)@ikg.uni-hannover.de

**Commission II, WG II/3**

**KEY WORDS:** Airborne Laser Scanning, Semantic Segmentation, Point Cloud, Height Normalisation, Sensor Independence

## ABSTRACT:

Semantic segmentation is one of the main steps in the processing chain for Airborne Laser Scanning (ALS) point clouds, but it is also one of the most labour intensive steps, as it requires many labelled examples to train a classifier. National mapping agencies (NMAs) have to acquire nationwide ALS data every couple of years for their duties. Having point clouds cover different terrains such as flat or mountainous regions, a classifier often requires a refinement using additional data from those specific terrains. In this study, we present an algorithm, which is able to classify point clouds of similar terrain types without requiring any additional training data and which is still able to achieve overall F1-Scores of over 90% in most setups. Our algorithm uses up to two height distributions within a single cell in a rasterized point cloud. For each distribution, the empirical mean and standard deviation are calculated, which are the input for a Convolutional Neural Network (CNN) classifier. Consequently, our approach only requires the geometry of point clouds, which enables also the usage of the same network structure for point clouds from other sensor systems such as Dense Image Matching. Since the mean ground level varies with the observed area, we also examined five different normalisation methods for our input in order to reduce the ground influence on the point clouds and thus increase its transferability towards other datasets. We test our trained networks on four different tests sets with the classes' ground, building, water, non-ground and bridge.

## 1. INTRODUCTION

Semantic segmentation is the task to assign every pixel in an image or every point in a point cloud a specific label, which describes its object class. As their duty, NMAs are required to acquire point cloud data covering their territory every couple of years. Classifying large amounts of point cloud data already causes some logistical problems. Despite that, NMAs are also dealing with point clouds, which cover different terrain types and consequently a different composition of point and object structures, which describe similar yet different domains. In addition, NMAs acquire not only ALS point clouds, but also derive very dense point clouds using aerial images. These point clouds contain different characteristics than ALS point clouds, which have been pointed out by Mandlburger et al. (2017). All these factors are to be considered when classifying point clouds on a nationwide level.

CNNs have been established as the state of the art method in several disciplines of remote sensing. There are three main approaches on point cloud classification using CNNs: point-wise, voxel-wise and raster-wise approaches. On the point-wise level, neighbouring points are collectively classified using automatically generated features from each point separately, but also as a set of points such as PointNet or its successor PointNet++ (Qi et al., 2017a; Qi et al., 2017b). Originally just tested on indoor point clouds and some CAD-generated point clouds, several extensions to adapt PointNet++ to terrestrial and aerial outdoor point cloud data have been proposed (Engelmann et al, 2017; Yousefhussien et al., 2018, Winiwarter et. al., 2019).

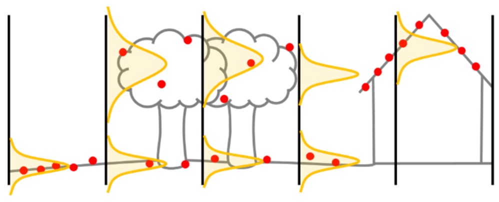
Figure 1. Schematic visualisation of our proposed algorithm using height distributions as input for classification. Gray: underlying objects, black: raster grid, red: measured points in the point cloud, yellow: calculated height distributions for each cell.

Other point-wise approaches consider the spatial relation between points using graph convolutional networks (Landrieu, Simonovsky, 2017; Te et al., 2018). The second approach of point cloud classification uses 3D voxels instead. Here, the spatial relationships of points are defined by the position within a set voxel grid. These voxel grids might be fixed around a point or along a globally defined raster and the trained CNNs propose a class value for each voxel, which is later projected back to the original point cloud (Huang, You, 2016; Tchampi et al., 2017; Schmohl, Sörgel, 2019). The last approach of point cloud classification using CNNs deals with point clouds on a raster-wise level. Here, the points are sorted in 2D raster cells and then used in a CNN like an ordinary image. While the input features vary from only using height values such as the minimal, mean and maximal height (Hu, Yuang, 2016; Politz, Sester, 2019) to some general features contained in ALS point clouds such as return numbers or intensity values (Zhao et al., 2018, Rizaldy et al., 2018). Some approaches even contain eigenvalue-based or normal vector based features (Xu, Yang, 2018).

* Corresponding author

At the end, the CNNs predict a class label for each raster cell, which is projected back onto the point clouds. However, most of the described classifiers either require ALS dependent attributes such as return numbers or intensity values, which are not available in image-based point clouds, or rely on extreme values such as minimal and maximal height to represent points in raster cells, which are prone to noise and very sensitive to specific heights.

In order to become independent of the sensor type, only the geometry of the point clouds or some derived characteristics from the geometry such as normal vectors can be used. If only the geometry of point clouds is concerned however, the individual height values can greatly differ for the same object, e.g. when the same objects are on flat or sloped terrain. Both, the flat and the sloped terrain, create two slightly different domains and to use a classifier from one domain on another, Domain Adaptation must be considered. There are already approaches for Domain Adaptation on aerial image data without requiring additional training data (Wittich, Rottensteiner, 2019). However, Domain Adaption may not even be necessary for point clouds as generic objects such as buildings, trees and bridges share similar geometrical characteristics independent of the underlying terrain. Only the underlying terrain such as flat plains or mountainous regions define those domain gaps between two different point cloud regions. Removing the ground influence from the point cloud geometry should already decrease or even close this domain gap. Consequently, determining the ground height and then subtracting it from the point cloud height should result in normalized point clouds, where a building originally close to sea level should have the same height and shape as a building in the mountains.

Several different approaches to generate a Digital Terrain Model (DTM) and consequently to approximate the ground level from point clouds have been made (Chen et al., 2017). Filter matrices or other mathematical functions are often used to approximate the ground surface (Zhang et. al., 2003; Zhang et al., 2016). Similarly, other approaches use additional information such as aerial images, intensity values from point clouds or a mix from geometry and ALS characteristics to determine the ground level within a point cloud (Gevaert et al., 2018; Yunfei et al., 2008; Rizaldy et al., 2018). However, an approximation of the ground on a small area patch such as a plane (Vosselman, 2013) may also reduce the ground influence enough to use a trained network on different domains and still achieve sufficient results.

The scientific contributions of this paper can be summarized as follows:

- We propose an algorithm, which classifies point clouds independent from sensor systems and point density, as it only requires the geometry of a point cloud as input.
- We explore and test five different height normalisation methods with varying level of complexity to remove the main part of the ground influence from the point cloud heights, while deducing what kind of complexity is necessary to achieve sufficient results using only the geometry.
- We test our trained CNN on four different datasets to prove the transferability of our approach on similar domains without requiring any additional training data.

The study is structured as follows. In section 2, we describe the calculation of our method (2.1), the different height normalisation methods applied on the point cloud (2.2), the network used for classification (2.3) as well as the loss function used for training the network (2.4).

```
Algorithm 1: Input generation for height distributions
Input: height values $z_i \in Z$ with $i = 1, \dots, L$ different height levels. $n_i \in N$ is the amount of points at height i with $N = n_1 + \ldots + n_L$.
Output: Either $(\bar{x}_0, s_0, \bar{x}_1, s_1)$ for two different height distributions or $(\bar{x}, s, \bar{x}, s)$ for one height distribution.
1:    sort(Z)
2:    $k^*, C_0, C_1 \leftarrow \mathrm{Otsu}_{1979}(Z)$
3:    $\bar{x}_0, s_0^2 \leftarrow$ mean and variance from $C_0 = \{z_i | z_i \leq k^*\}$
4:    $\bar{x}_1, s_1^2 \leftarrow$ mean and variance from $C_1 = \{z_i | z_i > k^*\}$
5:    $\bar{x}, s^2 \leftarrow$ mean and variance from Z
6:    $\lambda_0 \leftarrow N^{-1} \sum_{i=1}^{k^*} n_i$
7:    $\lambda_1 \leftarrow 1 - \lambda_0$
8:    $\mathrm{BIC}_{\mathrm{unimodal}} \leftarrow 2\ln(N) - 2\ln\left(\prod_{i=1}^{L} \mathcal{N}(z_i | \bar{x}, s^2)\right)$
9:    $\mathrm{BIC}_{\mathrm{bimodal}} \leftarrow$
      $4\ln(N) - 2\ln\left(\prod_{i=1}^{L} \sum_{j=0}^{1} \lambda_j \mathcal{N}(z_i | \bar{x}_j, s_j^2)\right)$
10:   if $\mathrm{BIC}_{\mathrm{unimodal}} \leq \mathrm{BIC}_{\mathrm{bimodal}}$ then
11:       return $(\bar{x}, s, \bar{x}, s)$
12:   else
13:       return $(\bar{x}_0, s_0, \bar{x}_1, s_1)$
```

Algorithm 1. Input generation for height distributions

Section 3 gives a short explanation of the training procedure and the different test sets and their respective differences. We show and discuss our results in section 4 and conclude this paper in section 5.

## 2. METHODOLOGY

### 2.1 Height Distributions

To be independent of the underlying point density or sensor system, the model requires a fixed resolution and a unique object representation. In order to achieve a fixed resolution, the point cloud is rasterized into raster cells with an edge length of 1m. In our approach, the unique object representation is based on the threshold selection algorithm from Otsu (1979). This algorithm allows partitioning the points within a raster cell depending on their height values into disjunct sets. These sets are then represented by up to j separate normal distributions using their empirical mean $\bar{x}_j$ and standard deviation $s_j$. We set j = 2, i.e. separate the points into a top and a bottom set. Depending on the object, the two distributions vary in shape as shown in Figure 1. For a raster cell with only ground pixels, the mean values will lie in the middle of the ground and $s_j$ will be small. However, for a raster cell with a tree and ground present, one distribution will ideally cover the ground points while the other will describe the tree with a much larger standard deviation than on the ground. Finally, buildings will have a similar top mean, but a smaller standard deviation than trees. Consequently, those objects can be separated. An overview of our algorithm is shown in Algorithm 1.

The original algorithm of Otsu (1979) searches for an optimal threshold $k^*$ to separate image histograms with L different grey values into foreground and background pixels. Instead of grey value histograms, we adapted the algorithm to split sorted height values into a bottom set $C_0$ and a top set $C_1$. Then, we calculate the empirical mean and variance $\bar{x}_0, s_0^2$ and $\bar{x}_1, s_1^2$ for the bottom and top distribution, respectively. Besides, we also calculate the empirical mean $\bar{x}$ and variance $s^2$ of all points within a raster cell. In order to decide, which model achieves the higher likelihood concerning the underlying data, e.g. one or two distributions, we use the Bayesian Information Criterion (BIC) (Schwarz, 1978).

The BIC is calculated as

$$\mathrm{BIC} = \ln(N)\,k - 2\ln(p(x|\hat{\theta}, M)) \tag{1}$$

with

$$p(x|\hat{\theta}, M) = \prod_{i=1}^{L} \sum_{j=0}^{J-1} \lambda_j \mathcal{N}(z_i \mid \bar{x}_j, s_j^2) \tag{2}$$

where:
- N: number of data points in the set
- k: number of parameters estimated by the model, either 2 for one or 4 for two distributions
- L: different height levels with $i = 1, \dots, L$
- J: number of distributions in the model
- $\lambda_j$: relative point frequency, $\sum_j \lambda_j = 1$.

The model with the lowest BIC is selected for the raster cell. If one distribution is preferred, the input for the network will consist of the mean and standard deviation calculated from all points $(\bar{x}, s, \bar{x}, s)$. For two distributions, the input will contain the values for the top and bottom distribution $(\bar{x}_0, s_0, \bar{x}_1, s_1)$. The majority class of the respective distribution decides the reference class. If a raster cell is empty, default values and an additional *no data* class are set. Finally, the raster data is split into non-overlapping raster images of size 100 x 100 m². This is large enough to still contain some ground information for normal sized residential and industrial buildings within a single raster image in most cases.

## 2.2 Normalisation Methods

Since the input only relies on the geometry itself, classifying other regions and consequently dealing with different ground heights is difficult. We examine several methods to determine the ground surface and thus obtain normalised height values for each point by calculating the height above ground. In addition, we trained and tested the network using all those different normalised point clouds as input to evaluate the influence of each normalisation method and its consequences on the classification result.

**2.2.1 Original:** Original height values are used for input generation, i.e. no normalisation is performed.

**2.2.2 DTM**: Since NMAs do collect point clouds every several years, we assume that a DTM or a labelled point cloud of the test region already exists. In this study, we applied a Delaunay triangulation on the labelled ground points for each test set to generate a surface model. The normalised point height is calculated by the distance between each point in the dataset and the plane from its closest triangle.

**2.2.3 LAStools:** The height above ground is calculated using the tool lasground_new from the software LAStools[1]. The tool only considers points from last pulse for its surface reconstruction. The default parameters are used for all datasets and the program returns the normalized height above ground.

**2.2.4 Local**: As already described in section 2.1, we split the data into an upper and lower set $C_1$ and $C_0$. Consequently, the bottom mean values $\bar{x}_0$ for each sample will mostly contain ground information. For each raster image, we sort the $\bar{x}_0$ values according to their height. Then, we take the lowest 10% of those values and calculate their mean $\bar{x}_{0,\mathrm{mean}}$.

In our experiments, 10% showed to be robust against noise as well as to be able to describe the lower parts of the ground quite nicely. $\bar{x}_{0,\mathrm{mean}}$ represents a local horizontal plane, which is subtracted from all points within the raster image to gain normalized points heights. In comparison to a local plane reconstructed from the minimal height value within a raster image, $\bar{x}_{0,\mathrm{mean}}$ is more robust towards noise.

**2.2.5 RANSAC:** Similar to 2.2.4, we exploit the split into an upper and lower distribution from the input generation and calculate the normalised height for each raster image separately. Instead of using the bottom mean values $\bar{x}_0$, we filter the point cloud, so that only the points from set $C_0$, which also have a standard deviation $s_0$ lower than 0.15m, remain. 0.15m equals the absolute standard deviation from most ALS point clouds and we discovered in our experiments, that ground points, but also building points usually tend to stay within this 0.15m boundary. Finally, we calculate a plane using a random sample consensus (RANSAC) algorithm on those filtered points. For the sampling process, we set the probability w, that a drawn observation belongs to the plane, at 0.6. Setting w at 0.6 enables the RANSAC algorithm to try more, possible solutions and consequently to find a suitable ground plane more likely. In addition, we set the desired probability z with which the model shall be found in the data to 0.95. The normalized height for a point within the sample equals the point to plane distance with the resulting plane from the RANSAC algorithm.

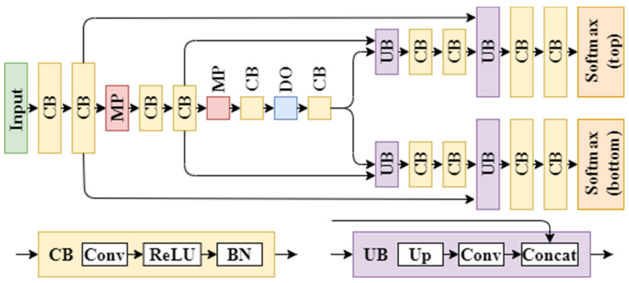

Figure 2. Network Architecture for the experiments using convolution blocks (CB), max pooling (MP), dropout (DO), up-sampling blocks (UB) and a softmax layer to output class probabilities for the top and bottom distribution, respectively.

## 2.3 Network Architecture

We extend the U-Net structure from Ronneberger et al. (2015) for the semantic segmentation of rasterised ALS point clouds. The network structure is shown in Figure 2. The encoder-decoder network structure allows an end-to-end training of the height profiles as described in section 2.1. The encoder has three levels and each level consists of two convolution blocks, where each block includes a convolutional layer using a 3x3-filter matrix, a Rectified Linear Unit (ReLU) as activation function as well as Batch Normalisation. Between levels in the encoder, max pooling is applied to down-sample the raster images by a factor of 2. In addition, the amount of calculated feature maps doubles in each level starting from 64 to 256. On the last level, there is a dropout layer with a dropout rate of 0.5 between the convolution blocks. After these blocks, the network splits into two separate, but identical built decoder parts. In order to restore the image resolution, up-sampling blocks are used. Up-Sampling blocks consist of an Up-Sampling layer using nearest neighbour interpolation, which is followed by a convolutional layer using a 2x2-filter matrix.

[1] https://rapidlasso.com/lastools/

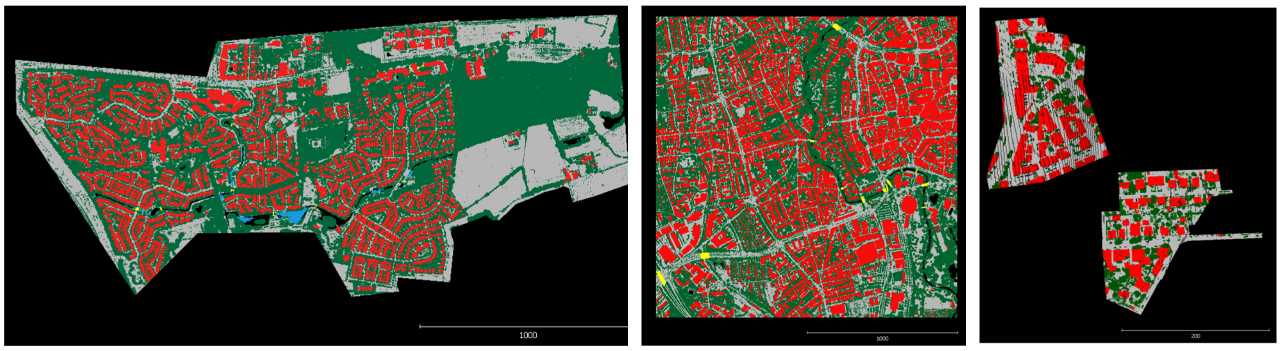

Figure 3. Reference classes for the AHN3 test set (left), Brunswick (middle) and Vaihingen (right). Grey: ground; green: non-ground; red: building; blue: water; yellow: bridge.

The Up-Sampling block concludes with concatenating the resulting feature maps from the convolution layer with those from the encoder of the same level. Like in the encoder, two convolution blocks follow the Up-Sampling block in each level. Finally, a softmax classifier predicts the class probabilities for each decoder part. One decoder predicts the class probabilities for the bottom, the other one for the top distribution. The network differentiates *ground*, *building*, *water*, *non-ground*, *bridge* and an additional *no data* class in order to deal with empty raster cells.

### 2.4 Loss Function

Since there are two outputs in the network, the loss function is the sum of two partial loss functions $L_j$. Each partial loss function $L_j$ is a weighted cross-entropy function

$$L_j = -\sum_{c=1}^{C} w_j(x) y_{c,j}(x) \log\left(\hat{y}_{c,j}(x)\right) \tag{3}$$

with the weight $w_j(x) = 1/f_{c,j}$, where $f_{c,j}$ is the relative class frequency, $y_{c,j}(x)$ is the reference and $\hat{y}_{c,j}(x)$ is the predicted label of class $c$ for distribution $j$. We also tested different weight functions. Nevertheless, those functions yielded worse predictions in our experiments.

## 3. DATA

In order to test the suitability of our method, we train the network from data of a single dataset (3.1) and apply it on several other datasets to test its generalization capability (3.1 - 3.4).

### 3.1 Rostock

The ALS point clouds provided by the NMA of Mecklenburg-Vorpommern (AFGVK) have a mean point density of 5 points/m² and cover an area of 20km² in southern Rostock, Germany in 2012. The mean ground level of the area varies from -1m to 56m. Urban areas with residential and industrial buildings in the North West and Nord East, garden plots with small cottages in between, huge agricultural areas and grassland in the South, forests, a river and several small lakes characterise the region (see Figure 4). The classes *ground* and *non-ground* are automatically created and manually controlled by the NMA. The classes *building*, *water* and *bridge* were manually added, but not fully controlled, so that some label error exist in the training data.

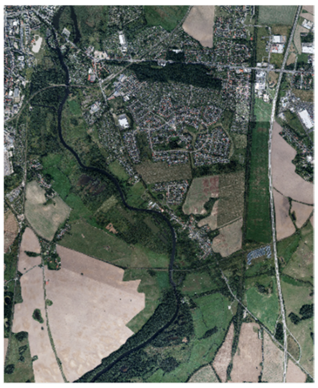

Figure 4. The training and test dataset of Rostock, Germany

The data is pre-processed as described in section 2.1, which results in 2000 raster images in total. 300 images are picked randomly for testing to ensure, that all different regions within the dataset are covered. To find the optimal hyperparameter for the network, the dataset is further split into five chunks for a 5-fold cross validation, where one is used as validation set and the remaining ones for training. The final network is trained on all 1700 training images.

### 3.2 Brunswick

The dataset is provided by the NMA of Lower Saxony (LGLN), Germany and covers an area of 4km² with a mean point density of roughly 8 points/m² of the city centre of Brunswick, Germany from 2015 (see Figure 3, middle). In contrast to the dataset in 3.1, this data contains a densely built-up area with various building shapes, which the network has never seen during training. The data is automatically labelled and manually checked into the classes *ground*, *building*, *non-ground* and *bridge*. Although the river Oker is flowing through the area, there are no natural water points contained in the dataset. The main ground level varies from 69m to 88m.

### 3.3 ISPRS Vaihingen 3D Semantic Labeling Challenge

We are also testing our method on the ISPRS 3D Semantic Labeling Contest[2] (Niemeyer et al., 2014). It contains two ALS point clouds for training and testing and covers an area of Vaihingen an der Enz, Germany in 2008. The mean point density is about 4 points/m² (8 points/m² on strip overlap) and the mean ground level varies from 265m to 288m. The point clouds contain the classes' *powerline*, *low vegetation*, *impervious surfaces*, *car*, *fence/hedge*, *roof*, *façade*, *shrub* and *tree*.

[2] http://www2.isprs.org/commissions/comm3/wg4/3d-semantic-labeling.html

We refrain from training or refining our network on the available training set, but apply our network as is on the test set (see Figure 3, right). In order to use the same network trained on the Rostock data of section 3.1, the classes *powerline*, *low vegetation*, *car*, *fence/hedge*, *shrub* and *tree* are condensed in the class *non-ground*. The class *impervious surfaces* is renamed as *ground* and the classes *roof* and *façade* are summarized to *building*. Compared to the other test datasets, the Vaihingen dataset has a much higher mean ground level, so it is possible to test the network on its generalization capability concerning different input heights and the influence of different applied normalisation methods.

### 3.4 Actueel Hoogtebestand Nederland (AHN3)

The Actueel Hoogtebestand Nederland (AHN3)[3] dataset covers most areas of the Netherlands by the end of 2019. The point density is about 16 points/m$^2$ and it is classified into the classes *unassigned*, which contains objects such as vegetation, vehicles and street furniture, *ground*, *building*, *water* and *bridge*. As the class *unassigned* contains the same objects as the class *non-ground* in the datasets of Rostock and Brunswick, we will further refer to it as *non-ground* as well. We test our network on a subset from tile C_33_FN1 of the city Deventer, Netherlands. Schmohl and Sörgel (2019) have already used this specific data for point cloud classification and the data is split into a training, validation and test set. As with the Vaihingen dataset, we refrain from training or refining our trained network using those datasets, but purely test our trained model on the described test dataset. The test dataset covers an area of 2.5km$^2$ with a mean ground level between 4 and 14m (see Figure 3, left). In contrast to the Rostock dataset, the bridges are mostly built for pedestrians and are consequently much smaller than bridges for vehicles, which are present in the Rostock and Brunswick dataset.

## 4. EXPERIMENTS

### 4.1 Training

The datasets are pre-processed according to the proposed method in section 2. Because of format issues, it was not possible to test Vaihingen on the LAStool normalisation setup. The proposed network is trained ten different times using the training data of the Rostock dataset. The final class predictions for all test sets are based on the maximal class probabilities of the network ensemble.

### 4.2 Classification Transferability

When used on different domains than the trained one, the results of any classifier are expected to become worse. However, the overall F1-Scores from the main domain (Rostock) to Brunswick or AHN3 decrease by only a small margin of about 1.5% and remain above 90% in most normalisation setups as shown in Table 1. Unfortunately, our approach is not able to achieve comparable results in the AHN3 test dataset to Schmohl and Sörgel (2019), who trained their classifier using the proposed training and validation datasets as described in section 3.4. With their approach, they achieved an overall F1-Score of 95.4% in the test set, which is roughly 5% higher than our results. However, our results still show a good transferability of the proposed approach onto datasets with similar terrain without the need of any additional training data, once the network is trained.

[3] https://downloads.pdok.nl/ahn3-downloadpage/

| F1-Score [%] | | Ground | Building | Water | Non-Ground | Bridge | Overall |
|---|---|---|---|---|---|---|---|
| Rostock | Original | 95.7 | 89.2 | 81.7 | 81.7 | 66.1 | 92.5 |
| | DTM | **96.3** | **91.3** | **92.3** | **83.0** | **86.3** | **93.4** |
| | LAStools | 95.7 | 90.2 | 71.7 | 82.2 | 66.2 | 92.5 |
| | Local | 95.8 | 90.7 | 80.3 | 82.1 | 85.4 | 92.7 |
| | RANSAC | 95.8 | 89.3 | 81.1 | 81.8 | 64.2 | 92.6 |
| Brunswick | Original | 22.7 | 52.0 | -- | 77.1 | 0.0 | 45.7 |
| | DTM | **94.1** | **95.4** | -- | **88.9** | **38.7** | **92.7** |
| | LAStools | 92.9 | 94.1 | -- | 88.3 | 4.8 | 91.6 |
| | Local | 92.6 | 93.6 | -- | 88.3 | 6.8 | 91.4 |
| | RANSAC | 92.5 | 93.2 | -- | 88.2 | 18.7 | 91.2 |
| Vaihingen | Original | 0.0 | 45.2 | -- | 0.0 | -- | 15.1 |
| | DTM | **95.7** | 85.7 | -- | **79.3** | -- | **86.9** |
| | LAStools | -- | -- | -- | -- | -- | -- |
| | Local | 87.5 | **90.3** | -- | 78.5 | -- | 85.4 |
| | RANSAC | 86.7 | 90.1 | -- | 78.7 | -- | 85.2 |
| AHN3 | Original | **93.3** | 88.5 | 9.6 | 88.8 | 0.0 | 89.8 |
| | DTM | **93.3** | **91.4** | **78.8** | **90.5** | 0.0 | **91.9** |
| | LAStools | 91.8 | 90.5 | 7.5 | 89.8 | 0.0 | 90.7 |
| | Local | 91.9 | 90.7 | 17.7 | 89.7 | 0.0 | 90.8 |
| | RANSAC | 91.9 | 89.7 | 18.0 | 89.4 | 0.0 | 90.5 |

Table 1. F1-Scores from the accumulated prediction of ten networks. Best results for each test set are marked bold.

Only the predictions of Vaihingen achieve lower overall F1-Scores of about 85% as shown in Table 1, which are mostly due to some issues at the border of the point clouds (see Figure 5). Results of the different test areas are shown in greater detail in Table 1 and Figure 5.

Even though the overall F1-Scores are similar in all test sets, the class specific F1-Scores vary notably. Having class weighting been used during training, classes with more available points such as *ground*, *building* and *non-ground* still achieve higher and more stable results in all test sets than the smaller classes *water* and *bridge* (see Figure 5, Table 1). As shown in Table 1, the *ground* class appears to be the most stable, but is also the most common class in all test sets. The best class F1-Scores for *ground* in each test set only differ by 3% with AHN3 achieving the lowest score of 93.3%. Similarly, *building* is quite stable and the best F1-Scores in each test set only vary between 90.3% and 95.4%. The later result is interesting as *building* was predicted better in Brunswick (95.4%) than in Rostock (91.3%). This might be due to the lack of vegetation close to most buildings in Brunswick, which consequently decreases the chance of having border issues between objects (see Figure 3, middle). Equally, one of the major drawbacks of *non-ground* classification originates in the limitations of our approach, which are further discussed in this section below. As a result, *non-ground* achieves best F1-Scores varying from 79.3% to 90.5% between test sets and roughly 2% difference within a test region. Water points hardly exist in ALS point clouds due to the reflection of the beam if the plane is not in nadir view to the respective water source. In most ALS point clouds, water points are usually spatially apart from the other object points and are mostly surrounded by the *no data* class representing the lack of points within a raster cell in our approach. As a result, the network learns this topological characteristic and thus classifies points on the riverside as *water*.

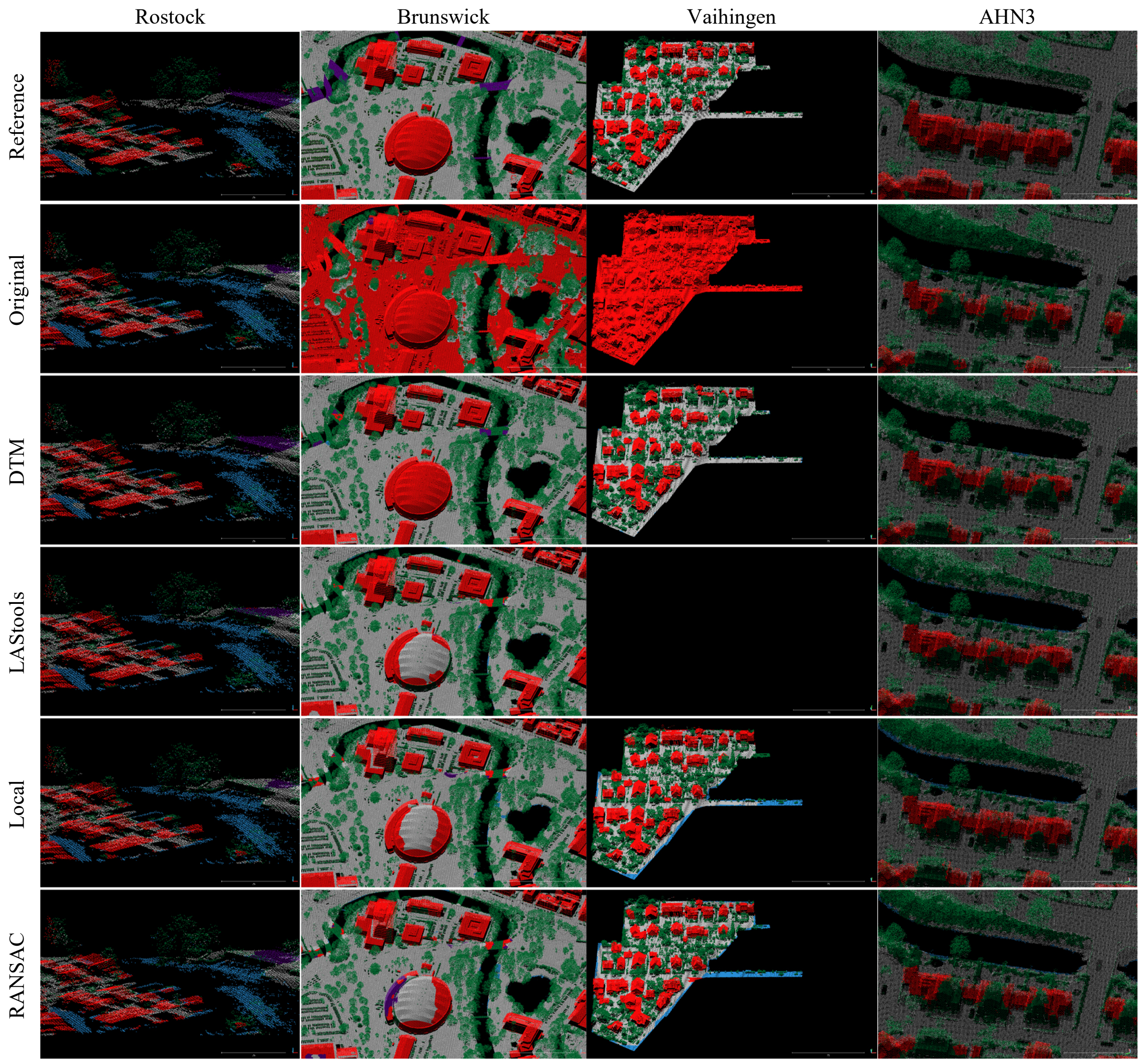

Figure 5. Exemplary results of the point cloud classification on all four test sets. Grey: ground; green: non-ground; red: building; blue: water; purple: bridge.

This is especially problematic, if the point cloud is small and the majority of a raster image is set to *no data* as found in the Vaihingen dataset (see Figure 5, Vaihingen). Here, the street going outwards and the border points of the whole test set are partially classified as *water* due to this problem resulting in poor overall F1-Scores as shown in Table 1. Another issue in all test sets are bridges, which are hardly classified at all except for the Rostock test set (see Table 1), but where their results still vary depending on the applied normalisation method. The training dataset only contains about a dozen bridges, which all were underpass bridges for vehicles. Consequently, it does not recognize footbridges, which are mainly present in the AHN3 dataset and which are much smaller than bridges for vehicles (see Figure 3, left). In Brunswick, where underpass bridges are present, the prediction differs with the normalisation method and is still quite below the results from the Rostock dataset. Although the kind of bridge is similar to the training dataset, only the trained networks using DTM or RANSAC normalisation are able to predict parts of the bridges correctly (see Figure 5, Brunswick). The location of these bridge parts is often at the foot of the bridge with a present slope in either the bridge or the ground below. When the bridge flattens, the main part of the bridge is classified either as *building*, when the bridge is above ground in the city, or *non-ground*, when the bridge is above a river. Consequently, it appears that the trained network ensemble is still able to differentiate and understand the context of its environment, even though the bridge itself is not detected as one.

Our approach performs the classification based on the number of distributions j within a raster cell. Consequently, only j classes are possible within each cell. This is an inherent limitation, which leads to some misclassification between object boundaries remaining in all tests. Due to the coarse resolution of 1m$^2$ in the raster cells, points of two neighbouring objects fall into the same raster cell. If the two objects belong to the same class, there is no issue unless the prediction of the network ensemble returns a wrong class in the first place. However, since the network ensemble only returns one class label per distribution, points within a mixed raster cell will automatically contain labelling errors when projected back to the original point cloud, since one of the objects is classified wrongly. This is especially the case for trees close to buildings (see Figure 5, Rostock and AHN3).

Decreasing the length of a raster cell, so less objects fall into the same raster cell, or increasing the amount of possible distributions for each raster cell, so more objects could possibly be predicted, might solve this issue. Equally, taking into account the entire class probabilities for a distribution instead of just the highest probability and adding a mapping process from these class probabilities to the points itself might also work. Another issue occurs by trying to split only a limited amount of points into two separate distributions. The Otsu algorithm finds a local minimum between a lower and upper distribution. However, in situations with ground and tree points, the few points within a raster cell are spread apart in height, so that there is no clear boundary between objects anymore. As a result, the algorithm fails and some tree points are assigned to the bottom distribution, which are then classified as *ground* (see Figure 5, Rostock). This is especially visible in the Rostock test dataset with only 5 points/m², which led to F1-scores for *non-ground* of only about 82% as shown in Table 1. This problem becomes less of an issue with a higher amount of points/m² like in the AHN3 set with 16 points/m² (see Figure 5). Independent of the normalisation method, it appears that some systematic errors during the training process happened, which result in some of the buildings being classified as *non-ground* (see Figure 5, AHN3).

## 4.3 Normalisation Methods

Independent of the specific test set, classification results show similar outcomes for the same normalisation method. The best results were achieved in all, but one case using an existing DTM as shown in Table 1. Considering no normalisation and consequently using the original height as input yields the worst classification results. With a higher mean ground level like in the Brunswick and Vaihingen test set, the ensemble of networks tend to classify all points as *building* and *non-ground*, which have typically higher z values than ground (see Figure 5). This is not surprising as a correct reconstruction of the ground ensures similar heights of objects independent of the specific region, which makes classification simpler. Although the normalisation using LAStools generates a similar height above ground as the DTM normalisation, the overall results of this method are more comparable with the local and RANSAC normalisation (see Table 1). In contrast, when the smaller classes *water* and *bridge* are concerned, the results with LAStools are even worse than the other two methods. These misclassifications using LAStools are mostly due to some reconstruction errors on the edges of tiles as well as at areas, where buildings are recognized as ground by mistake. The normalisation methods with a local plane as well as using RANSAC yield very similar results as shown in Table 1. While RANSAC classifies *water* by a small margin better than the local plane, *bridge* varies with the test sets. For the larger classes *ground*, *non-ground* and *building*, the local plane normalisation yield slightly better results (see Table 1).

There are some advantages and disadvantages regarding certain normalisation methods. Only the network ensemble using DTM normalisation results in complete and correct predictions on large and flat or very complicated roof shapes, while the other methods fail partially (see Figure 5, Brunswick). However, in order to use this method, a DTM or given labelled point cloud of the same region is required, which is not available in most practical circumstances. Likewise, the approach using LAStools requires the point cloud to have information about the last pulse. However, point clouds from different sensor systems, e.g. from images, do not contain such attributes. In addition, the results from the network ensemble using a LAStools normalisation were, at least in our experiments, on the same level or below the other normalisation methods excluding using the original height. The network ensemble using the original height as input appears to yield poor classification results, when the mean ground height deviates a lot from the one used for training the network. Consequently, it could still be used for areas with similar topography as the training set as shown with the Rostock and AHN3 test set (see Figure 5). The most simple normalisation methods using a local plane or RANSAC do not require any additional information other than the geometry and yield results, which are less than 2% worse in the overall F1-Score when compared to the DTM normalisation approach in all test sets. By removing the majority of the ground influence using either the local plane or RANSAC, the results compared to using the original height improve greatly. Likewise, it appears that small local curvatures or slopes in the ground do not affect the classification process as much, but rather the different mean ground height in general. However, this is only valid for flat terrain, where the local plane and RANSAC normalisation achieve similar normalized heights. When tested on terrain with higher inclination, the normalisation method with RANSAC might surpass the one with a local horizontal plane, as this will no longer match the main ground surface anymore.

## 5. CONCLUSIONS

In this work, we presented a geometry-based classification approach, which uses height distributions calculated from a rasterized set of points. Since it only requires the geometry, the approach is independent of the sensor system and is applicable for different point densities as long as there are enough points to calculate proper height distributions within a raster cell of given size. We trained our network on one dataset and tested it on four different test sets without the use of any additional training data to show its transferability towards slightly different ground heights. Since all test sets contain flat terrain, the transferability on mountainous terrain remains open for future work. We tested five different height normalisations to reduce the ground influence on the point cloud heights greatly. Our experiments showed that normalisation using a given DTM achieves best results. As such, a DTM is often not available and comparable alternatives are necessary. Simpler normalisation methods using a local horizontal plane or an oblique plane calculated by a RANSAC algorithm also resulted in good class predictions without requiring any additional data source.

However, there are still some issues remaining. Although class weighting has been applied during training, the classification of smaller classes such as water and bridge often result in misclassifications. Due to the sparsity of water points, the network associates water along point cloud borders and thus classifies riversides among others as water. Bridges cause various problems as they are difficult to normalize, require a complete training data set with different kinds of bridges and are hardly available in point clouds due to their sparsity in terrain. Consequently, our approach was not able to classify them in a satisfying manner. We discussed several ways to improve our proposed approach in future works.

## ACKNOWLEDGEMENTS

The Vaihingen dataset was provided by the German Society for Photogrammetry, Remote Sensing and Geoinformation (DGPF) [Cramer, 2010]: http://www.ifp.uni-stuttgart.de/dgpf/DKEP-All.html.

We thank the Landesamt für Geoinformation und Landesvermessung Niedersachsen (LGLN), the Amt für Geoinformation, Vermessungs- und Katasterwesen (AFGVK) and the Landesamt für Vermessung und Geoinformation Schleswig-Holstein (LVermGeo) for providing the point cloud data and for their support of this project.

## REFERENCES

Chen, Z., Gao, B., Devereux, B., 2017. State-of-the-Art: DTM Generation Using Airborne LIDAR Data. *Sensors* 17 (1), 150, https://doi.org/10.3390/s17010150.

Cramer, M., 2010. The DGPF test on digital aerial camera evaluation – overview and test design. *Photogrammetrie – Fernerkundung – Geoinformation*, 2, 73-82.

Engelmann, F., Kontogianni, T., Hermans, A., Leibe, B., 2017. Exploring Spatial Context for 3D Semantic Segmentation of Point Clouds. *ICCVW 2017.*

Gevaert, C.M., Persello, C., Nex, F., Vosselman, G., 2018. A deep learning approach to DTM extraction from imagery using rule-based training labels. *ISPRS Journal of Photogrammetry and Remote Sensing*, 142, 106-123.

Huang, J., You, S., 2016. Point Cloud Labeling using 3D Convolutional Neural Network. *ICPR 2016.*

Landrieu, L., Simonovsky, M., 2018. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs. *CVPR 2018.*

Niemeyer, J., Rottensteiner, F., Soergel, I., 2014. Contextual classification of lidar data and building object detection in urban areas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 87, 152-165.

Otsu, N., 1979. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9 (1), 62-66.

Politz, F., Sester, M., 2019. Joint classification of ALS and DIM point clouds. *ISPRS Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, XL-II-2/W13, 1113-1120.

Qi, C., Su, H., Mo, K., Guibas, L.J., 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CVPR 2017.*

Qi, C., Yi, L., Su, H., Guibas, J., 2017b. PointNet++: Deep Hiarchical Feature Learning on Point Sets in a Metric Space. *NIPS 2017.*

Rizaldy, A., Persello, C., Gevaert, C., Elberink, S.O., Vosselmann, G., 2018. Ground and Multi-Class Classification of Airborne Laser Scanner Point clouds Using Fully Convolutional Networks. *Remote Sensing*, 10 (11), 1723.

Ronneberger, O., Fischer. P., Brox, T., 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention 9351. Springer, LNCS, 234-241.

Schmohl, S., Sörgel, U., 2019. Submanifold sparse convolutional networks for semantic segmentation of large-scale ALS point clouds. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, IV-2/W5, 77-84.

Schwarz, G., 1978. Estimating the dimension of a model. *The Annals of Statistics*, 6 (2), 461-464.

Tchapmi, L.P., Choy, C.B., Armeni, I., Gwak, J., Savarese, S., 2017. SEGCloud: Semantic Segmentation of 3D Point Clouds. *International Conference of 3D Vision (3DV) 2017.*

Te, G., Hu, W., Zheng, A., Guo, Z., 2018. RGCNN: Regularized Graph CNN for Point Cloud Segmentation. *ACM international conference on Multimedia 2018*, 746-754.

Vosselman, G., 2013. Point cloud segmentation for urban scene classification. *ISPRS Int Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, XL-7/W2, 257-262.

Winiwarter, L., Mandlburger, G., Schmohl, S., Pfeifer, N., 2019. Classifiation of ALS Point Clouds Using End-to-End Deep Learning. *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 87 (3), 75-90.

Wittich, D., Rottensteiner, F., 2019. Adversarial Domain Adaptation for the classification of aerial images and height data using Convolutional Neural Networks. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, IV-2/W7, 197-204.

Xu, Z., Yang, Z., 2018. Eigenentropy based Convolutional Neural Network based ALS point clouds classification method. *ISPRS Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, XLII-3, 2017-2022.

Yousefhussien, M., Kelbe, D. J., Ientilucci, E. J., Salvaggio, C., 2018. A multi-scale fully convolutional network for semantic labeling of 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 143, 191-204.

Yunfei, B., Guoping, L., Chunxiang, C., Xiaowen, L., Hao, Z., Qisheng, H., Linyan, B., Chaoyi, C., 2008. Classification of Lidar point cloud and generation of DTM from lidar height and intensity data in forested area. *ISPRS Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, XXXVII (Part B3b), 313-318.

Zhang, K., Chen, S., Whitman, D., Shyu, M., Yan, J., Zhang, C., 2003. A progressive morphological filter for removing nonground measurements from airborne LIDAR data. *IEEE Transactions on Geoscience and Remote Sensing*, 41 (4), 872-882.

Zhang, W., Qi, J., Wan, P., Wang, H., Donghui, X., Xiaoyan, W., Yan, G., 2016. An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote Sensing*, 8 (6), 501.

Zhao, R., Pang, M., Wang, J., 2018. Classifying airborne LiDAR point clouds via deep features learned by a multi-scale convolutional neural network. *International Journal of Geographical Information Science*, 32 (5), 960-979.