Region-based Edge Convolutions with Geometric Attributes for the Semantic Segmentation of Large-scale 3D Point Clouds

Jhonatan Contreras, Sven Sickert, and Joachim Denzler

***Abstract*—In this paper, we present a semantic segmentation framework for large-scale 3D point clouds with high spatial resolution. For such data with huge amounts of points, the classification of each individual 3D point is an intractable task. Instead, we propose to segment the scene into meaningful regions as a first step. Afterward, we classify these segments using a combination of PointNet and geometric deep learning. This two-step approach resembles object-based image analysis. As an additional novelty, we apply surface normalization techniques and enrich features with geometric attributes. Our experiments show the potential of this approach for a variety of outdoor scene analysis tasks. In particular, we are able to reach 89.6% overall accuracy and 64.4% average intersection over union (IoU) in the Semantic3D benchmark. Furthermore, we achieve 66.7% average IoU on Paris-Lille-3D. We also successfully apply our approach to the automatic semantic analysis of forestry data.**

***Index Terms*—Geometric Deep Learning, 3D Point Clouds, Semantic Segmentation, Outdoor Scenes.**

I. INTRODUCTION

In recent years, companies and research groups have increased their interest in the use of *Light Detection and Ranging* (LiDAR) scanners. These scanners can be used to generate precise spatial information about the shape, surface, and other geometric characteristics of occurring objects in a scene. LiDAR uses pulsed beams of light to measure distances from a scanner to the surface of objects in a scene. The result is typically stored as a 3D point cloud. Thus, the data is composed of a collection of non-uniformly distributed points in a continuous space (x-, y-, z-coordinates), which can be referred to as unstructured data. In some LiDAR campaigns, images are captured simultaneously to retrieve additional color or reflectance information of objects in a scene. Merging such information creates more meaningful point clouds.

Especially in geological sciences, a setup with LiDAR offers several advantages over regular 2D images. In dense forest areas, for instance, an aerial 2D photograph fails to capture the terrain surface in areas with thick canopy cover. Furthermore, clouds can make satellite imaging difficult. On the other hand, terrestrial cameras fail to capture scenes completely due to occlusions by nearby objects in the line of sight of the camera. In poorly illuminated scenes or during darkness, a 2D image captures only a few or very noisy information. In contrast, LiDAR is an active sensor and can collect data during both day and night. Although handling such data is not as intuitive as processing images, point clouds are used in numerous applications. Examples include the generation of canopy models [1], individual tree segmentation [2], building models, digital terrain elevation models [3], or semantic understanding of forest and urban areas. The latter can be achieved via semantic segmentation. It aims at assigning one label from a set of pre-defined classes to each point of a point cloud [4], [5]. Nearby points of the same class form semantically meaningful segments that resemble real-world object boundaries. For instance, in an urban outdoor scene, the classes could include natural terrain, vegetation, buildings, and cars. Semantic segmentation is an essential intermediate step towards complex tasks such as autonomous car driving, urban planning or disaster prevention and mitigation. It is crucial for automatic decision making.

For large high-resolution outdoor scenes, point-wise classification approaches are an intractable problem. However, the semantic segmentation of a scene can also be achieved differently. To reduce the complexity of the task, points can be grouped into segments before classification. Most commonly, such segments are voxels in regular volumetric grids [6], [7], [8]. This process reduces the complexity and, thus, computational requirements. However, at the same time, it decreases the output resolution. Furthermore, especially in outdoor scenes, a large number of voxels are empty.

Finally, rectangular grids are not a natural division for real-world scenes. In most cases, a unit contains points of different semantic classes. A consistent global labeling of all points in such voxels will likely lead to many misclassifications. It can be substituted by unsupervised segmentation, which obtains 3D segments based on visual or geometric criteria. In the best-case scenario, the resulting 3D segments are consistent with the spatial geometry and do not cross object boundaries. Thus, 3D segments are a more natural representation of point groups than voxel grids. At the same time, the quality of the initial segmentation method can have considerable effects on the behavior of subsequent processing steps.

In this paper, we propose a method based on PointNet [9] using a geometric deep learning operation called edge convolution [10]. It can capture local geometric attributes of adjacent segments. As an initial segmentation step, we follow

Manuscript submitted: February 2020.
Manuscript revised: May 2020.

J. Contreras is with Citizen Science Lab at DLR Institute of Data Science Jena and Computer Vision Group, Friedrich Schiller University Jena, Germany (e-mail: jhonatan.contreras@uni-jena.de).

S. Sickert is with Computer Vision Group, Friedrich Schiller University Jena, Germany (e-mail: sven.sickert@uni-jena.de).

J. Denzler is with Computer Vision Group, Friedrich Schiller University Jena, Germany (e-mail: joachim.denzler@uni-jena.de).

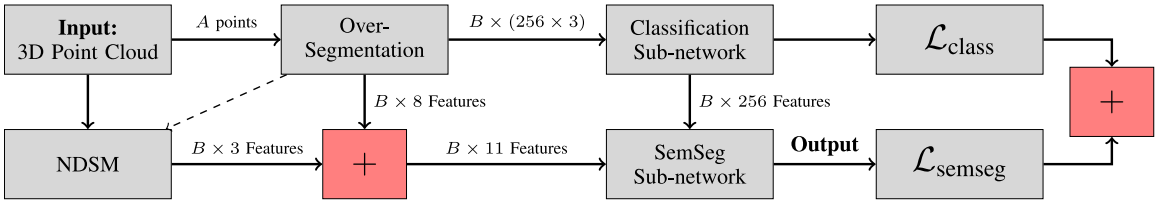


Fig. 1: Pipeline of our approach for semantic segmentation: The original point cloud consists of A points, which is typically several orders of magnitude larger than the number of segments B. Segments are generated using an unsupervised segmentation approach. Based on the input and the unsupervised over-segmentation, NDSM is a means of modeling additional geometric attributes. These are added to the segment's features and passed to a semantic segmentation sub-network. The losses of both the global classification and local semantic segmentation sub-network are combined for optimization.

the approach proposed by [11]. For the semantic segmentation of outdoor 3D scenes, we propose to use the normalized elevation to simplify the distinction between elevated and non-elevated objects. In urban scenes, for instance, it benefits the distinction between ground, high vegetation, and building structures. Our experiments demonstrate the viability of our approach for both high-resolution and low-resolution point clouds displaying outdoor scenarios.

The remainder of this paper is organized as follows. In Sec. II we report on related work and put our approach in context to the state-of-the-art. After that, in Sec. III we describe our approach based on convolutional networks using edge convolutions and a *Normalized Digital Surface Model* (NDSM) for 3D point cloud data. Sec. IV follows with an extensive experimental evaluation of our approach. We use diverse and challenging outdoor datasets that differ in both quality and the underlying recording settings. In Sec. V we summarize our findings.

II. Related Work

There are mainly three kinds of approaches to directly process 3D point cloud data: point-wise, voxel-wise, and segment-wise. Alternatively, a point cloud data can be mapped into two-dimensional space and processed accordingly. The output is then obtained by applying standard image CNNs on multiple 2D images views [12], [13], [14]. In the following, we will put our proposed approach in relation to these works.

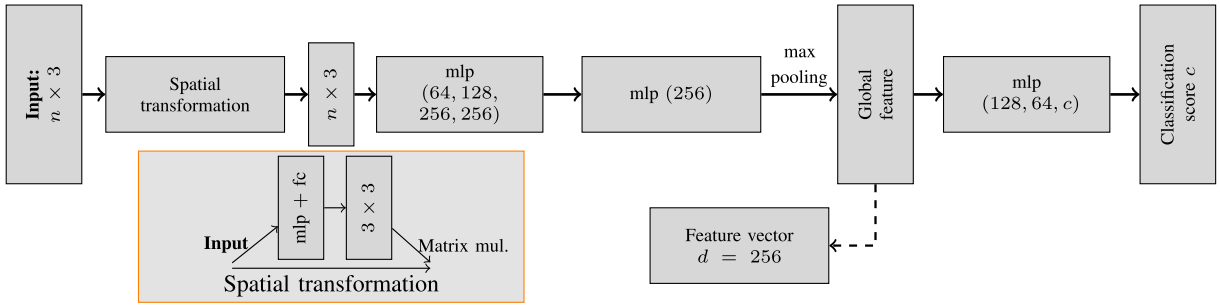
A. Voxels-wise Approaches

VoxelNet proposes a generic 3D detection learning network that unifies feature extraction and bounding box prediction into a single stage. Selected points inside each voxel are transformed using a proposed voxel feature encoding [6]. A region proposal network [15] uses the previous output to generate detections and bounding boxes. The authors of *OctNet* [7] create a hierarchical partition of the 3D space, that exploits its sparsity characteristics using a set of unbalanced octrees [16]. It focuses on dense regions, obtaining more partitions only over the relevant dense regions without affecting the resolution and accuracy. Their 3D CNN redefines traditional operators for convolution and pooling, to make deep learning tractable for high-resolution inputs. In [8], a technique is proposed, where each voxel is encoded using only 1 Bit, which saves computational time and effort. An essential contribution is the implementation of a lightweight CNN model, which obtains a low-power and low-cost inference targeting robots, drones, and cars. It is independent of the number of points and their distributions inside of voxels. However, the voxelization process leads to a loss of information and details in both geometric and visual aspect.

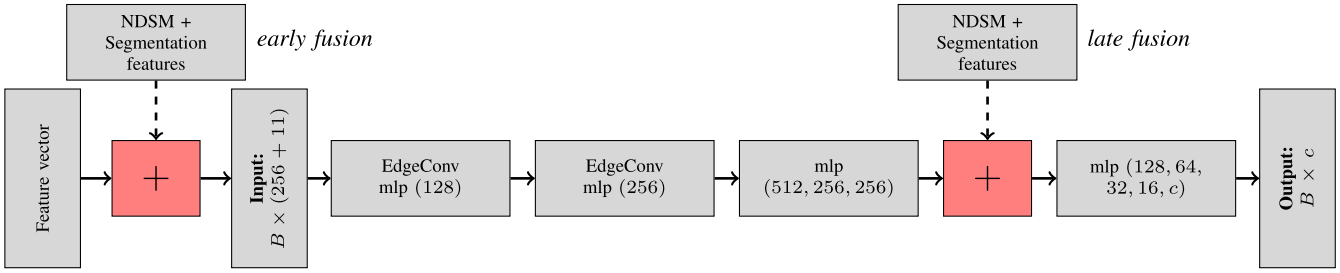
B. Points-wise Approaches

Point-wise approaches are computationally expensive but can offer high details in scenarios with low point densities. The authors of [17] propose to compute 3D moment invariant features for each occurring point. They are invariant under scaling, rotation, and translation. Additionally, the feature representation is augmented by local contextual information, which is generated using cascaded classification. Context features are shared among nearby points. Another example of hand-crafted features is the work on SHOT descriptors [18]. The surrounding of each point is organized in a structured spherical neighborhood with bins. Based on the distribution of nearby points, a meaningful histogram can be created, which is invariant under rotation. A fast semantic segmentation point-wise approach is proposed in [19], where a Random Forest classifier is used. The points are described by a set of features of three different types. First, they use a subset of geometric features based on eigenvalues and corresponding eigenvectors. Additionally, the first and the second-order moments of the point neighborhood around the eigenvectors are integrated. The third type of feature comprises height values (z-coordinates) and vertical range.

In contrast to these works on features design, recent point-wise classification approaches focus on the use of deep neural networks [9], [10]. *PointNet* [9] divides a scene into 3D blocks. Points inside a block are then aligned by a spatial transformer network (STN) [20] to retrieve point features. A max-pooling layer serves as a symmetric function to aggregate information from all the points resulting in a shared global feature, which is invariant to input permutation. Finally, concatenated global and local features are used to predict a class score for each point. Follow-up work [21] addresses missing local context information by applying a hierarchical PointNet variant called PointNet++. The authors of PointCNN [22] propose a hierarchical convolution method with an X-Conv module that aggregates inputs into less points with more valuable features.



(a) Classification Sub-network



(b) Semantic Segmentation Sub-network

Fig. 2: Architectural details on the sub-networks used in our pipeline: (a) **The classification sub-network** takes as input n 3D points from the segments. A max-pooling operation aggregates all information in a 1D global feature descriptor, which is used as input for the segmentation Sub-network. The spatial transformation block is used to align an input point cloud to a canonical space by applying a learned 3×3 transformation matrix. (b) **The Semantic segmentation sub-network** takes as input the 1D global feature descriptor and information extracted in previous steps or from other sources. Those additional features can be added at an early or late step in the system. The network computes features by applying edge convolutions and multi-layers perceptron until the last layer classifies each segment in one of c classes.

Also motivated by the individual classification of points, the authors of [10] adapt ideas from CNNs to incorporate neighborhood information. In particular, an edge convolution operation based on graphs is introduced. It allows to find semantically similar geometric areas within a point cloud. PointSift [23] introduces a module that can be incorporated to most of the existing PointNet based methods to improve the permutation invariance. This method applies a scale-invariant feature transform to capture a feature representation of the points. The authors of [24] propose a multi-scale point-wise CNN based on PointNet. It consists of the four components 3D convolutions, up- and down-sampling, dynamic feature extraction, and post-processing using conditional random fields (CRF). Furthermore, the authors of [25] present a module called PointGCR that can be included in 3D CNNs to incorporate semantic context dependencies information with global reasoning. It performs graph convolution operations considering each channel of an output layer as a graph node, while their dependencies are modeled via edges.

In another work, the authors of [26] propose a network called MHNet, which is composed of four connected down-sampling modules using PointNet layers to capture local features at multiple scales. The point cloud is up-sampled, concatenating the local features obtained in the four stages of the hierarchical network. Additionally, a CRF is applied within the output layer as post-processing step. TGNet [27] is a geometric graph convolution network applied on multiscale neighborhoods that learns expressive and compositional local geometric features. Their filters are defined as the products of the local point features and the neighboring geometric features where Gaussian weighted Taylor kernels represent the geometric features. The work on MS-PCNN [24] proposes an end-to-end feature extraction framework inspired by PointNet and edge convolutions. Its global and local features are extracted using dynamic edge convolutions on points at different scales utilizing down-sampling and up-sampling modules. The previously mentioned multiscale methods and others such as 3P-RNN [28] have the advantage of including information from a larger neighborhood. Using the neighborhood size the influence of context can be controlled and scale invariance can be improved.

C. Segments-wise Approaches

Another way to achieve a semantic segmentation is to classifiy segments instead of individual points. The authors of [11] propose to combine PointNet with graphical models and unsupervised segmentation. The whole point cloud is divided into geometrically homogeneous segments. PointNet is utilized to obtain a global feature for each of those segments. A super-point graph is built by modeling segments as graph nodes on which a graph convolution can be applied. The authors of [29] utilize a supervoxel-based method to generate nearly uniform segments. For each segment, attributes are generated, then a Random Forest classifier creates an initial label, which is re-

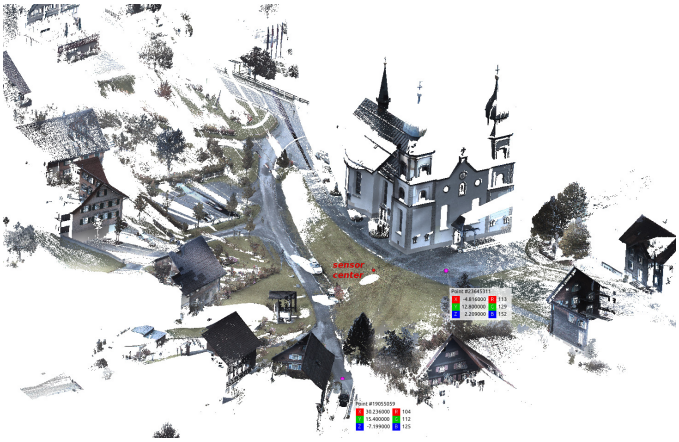


Fig. 3: The *Bildstein station 5* scene shows an urban area in Semantic3D dataset [30] without surface normalization. The coordinate system corresponds to the sensor position (red dot). We can observe the coordinates for two points on the asphalt with values $z = 2.2m$ and $z = -7.19m$.

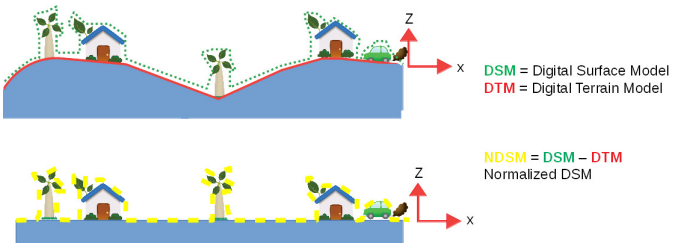


Fig. 4: The Digital Surface Model (DSM) represents earth's surface and includes all objects on it. In contrast, DTM is a representation of a terrain's surface without any objects. In the NDSM the terrain is normalized to zero, which allows direct measurement of object heights.

fined using a supervised graph-based model at the end. Similar to [11], our proposed method uses unsupervised segmentation to split the scene in a first step. However, in our approach, we capture local information of the points in each segment using an additional sub-network. We make use of extensive local knowledge, based on geometrical attributes of the neighboring segments through edge convolutions. Thus, our approach aims to reduce complexity and memory consumption combining ideas of [11] and [10] to overcome the limitations of global features of the PointNet approach [9].

III. Deep Learning using Attributes and Elevation Modelling

Our proposed deep learning-based framework can manage large-scale point clouds of outdoor scenes with high spatial resolution. To achieve this goal, we segment a scene by grouping geometrically and visually similar points together. Next, our network classifies segments instead of individual points using an architecture similar to PointNet. In Sec. III-A, we describe our whole pipeline in detail. The modeling of additional geometric information features is covered in Sec. III-B. Afterwards, we recapture edge convolutions in Sec. III-C and how they are used in our case. Finally, we propose an efficient organization of the input data in Sec. III-D

A. Pipeline and Attributes

An overview of our whole approach is given in Fig. 1. In the first step, the *Normalized Digital Surface Model* (NDSM) of the scene is computed, stored, and used as part of the input in the fourth step. We will come back to this model later in Sec. III-B. Simultaneously, an unsupervised segmentation based on [11] is performed to reduce the complexity of the data. We selected this approach, as the size of the segments depends mainly on the local geometric homogeneity. In this way, small segments can be obtained for objects such as bollards or trash can, as well as large segments for uniform surfaces such as roads or walls. Typically, a high-resolution scene can contain millions of points. The analysis of all individual points is computationally expensive and sometimes even redundant. In Fig. 5b and Fig. 7b we visualize segments created by unsupervised segmentation.

As a result, instead of working with millions of points, our approach needs to analyze several thousand segments. Each of them contains a discrete subset of points with a fixed number of elements. We accomplish segmentation using a particular set of attributes computed for each point. Nearby points with similar characteristics should be part of the same group. Those attributes include *linearity*, *planarity*, *scattering* and *verticality* [19]. They are used again in the classification step as additional feature input.

In the second step, each segment is considered as an independent 3D shape containing n points in $\mathbb{R}^3$ with x-, y- and z-coordinates. These shapes are used as input to our classification sub-network based on PointNet [9]. The latter is composed of two multi-layer perceptron blocks separated by an aggregation function, which combines the information from all points in the segment (see Fig. 2a). The output of the aggregation function is a feature vector, which serves as input for the semantic segmentation sub-network. Thus, the feature vector includes a set of attributes, which are invariant under permutation and spatial transformations that characterize segment properties.

In the following step, NDSM is applied to augment the feature vector with minimum, maximum, and average elevation for each segment. A more detailed explanation follows in Sec. III-B. Additionally, a segment's length, volume, surface, and the number of points are determined and added as attributes. As a result, the feature vector contains eleven additional attributes. The feature vector is used as input in the semantic segmentation sub-network, which applies edge convolution operations (see Sec. III-C and Fig. 2b) to convolve segments considering a local neighborhood. Finally, a joint architecture is applied between both networks in an end-to-end learning process. An output score is computed for each segment.

B. Normalized Digital Surface Model

Terrain elevation may vary a lot in forestry areas. In urban areas, the terrain most often is flat or has only fluctuation

TABLE I: An overview of the attributes we use to enrich the feature vector of each segment in the semantic segmentation sub-network. The majority of them are taken from [31], where a more detailed explaination can be found.

Attribute	Definition
Normalize z min Normalize z max Normalize z mean	NDSM
Linearity	$\frac{\lambda_1 - \lambda_2}{\lambda_1}$
Planarity	$\frac{\lambda_2 - \lambda_3}{\lambda_1}$
Scattering	$\frac{\lambda_3}{\lambda_1}$
Verticality	$\sum_{j=1}^{3} \lambda_j \lvert[u_j]_i\rvert$
Length	λ_{s1}
Surface	$\lambda_{s1} \cdot \lambda_{s2}$
Volume	$\lambda_{s1} \cdot \lambda_{s2} \cdot \lambda_{s3}$
Number of points	$\lvert S_i\rvert$

in altitude. Fig. 3 shows a scene of the Semantic3D dataset, which we later use in our experiments. The LiDAR sensor is located over a small hill, and it is represented as a red dot. The coordinate system corresponds to the sensor position. Thus, the points above it have positive z values, while points below it have negative z values. Fig. 3 shows two points, where the altitude difference is greater than nine meters on the ground level. In the absence of global geo-information for calibration, the LiDAR sensor is the center of the reference system. Consequently, a normalization of the surface is desirable.

Additionally, in point cloud datasets, pairs of classes like high and low vegetation or building and hardscape can have small inter-class variances. For instance, in [30], the class for low vegetation includes flowers and small bushes. On the other hand, the high vegetation class includes trees and large bushes. The two are separated using a simple threshold value of two meters. In another example, garden walls belonging to the hardscape class pose similar geometric attributes to building walls. However, the latter are much taller than garden walls.

To allow direct measurement of object heights, we define a set of ground models. They also help in simplifying the distinction between elevated and non-elevated objects. The *Digital Surface Model (DSM)* represents the earth's surface and includes all objects on it. In contrast to DSM, the *Digital Terrain Model (DTM)* is a representation of a terrain's surface without any objects. Typical objects include cars, plants, and buildings. The DSM can be directly obtained as the outer hull of the point cloud or terrain's elevation data (Fig. 4). Finally, the *Normalized Digital Surface Model (NDSM)* provides valuable information in which the terrain is everywhere set to a standard of zero. It is generated by subtracting the digital terrain model from the digital surface model as

$$NDSM = DSM - DTM \ . \tag{1}$$

Several filtering algorithms have been developed in the last decades. In our approach, we use the progressive morphological filter (PMF) proposed in [32] to distinguish between ground and non-ground points. Afterward, we define a radius

TABLE II: Number of points and number of segments for examples of the *Paris-Lille-3D* dataset [35].

Training data	Points (Millions)	Segments
Lille 1.1	29.8	4681
Lille 1.2	29.7	4469
Lille 2	21.2	2605
Paris	37.3	8936
Test data		
Dijon 9	10.0	988
Ajaccio 2	10.0	1083
Ajaccio 57	10.0	1305

and use an interpolation algorithm to complete the DTM. We selected PMF as it is implemented in the GDAL open source library. It is convenient, acceptably fast, and we acquired satisfactory terrain models. However, there are alternatives like cloth simulation filter [33], that could be used, as well. In fact, in some cases, surface models are already available from other sources like satellite LiDAR data. From NDSM, we can extract the maximum, minimum, and average elevation of the points within a segment. These attributes represent extra information to enrich a segment in addition to geometric attributes. Finally, all of the previously mentioned attributes are added to the feature vector that serves as input for our semantic segmentation sub-network.

C. Edge Convolution for Segments

In the following, we adopt the edge convolution operation introduced by the authors of [10] as it is an essential part of our pipeline. Please note, edge convolutions are originally defined for points. We extend them to be used in conjunction with segments and consequently within our sub-network for semantic segmentation (see Fig. 2b).

As a prerequisite, we first introduce the notations which will be used throughout this section. $C = \{c_1, ..., c_n\}$ denotes a F-dimensional point cloud consisting of n points, where $C \subseteq \mathbb{R}^F$. Thus, in the case $F = 3$, each element contains 3D coordinates $c_i = (\bar{x_i}, \bar{y_i}, \bar{z_i})$. This representation can be extended to include additional information representing color, verticality, scattering, among others.

We use the unsupervised segmentation method called geometric partition with global energy defined in [11]. It computes the point cloud partition using a set of d_g geometric features $f_i \in \mathbb{R}^{d_g}$ for each point c_i. In particular, the set of feature d_g includes linearity, planarity, scattering, and verticality. According to [31] those attributes are defined using the eigenvalues $\lambda_1 \geqslant \lambda_2 \geqslant \lambda_3$ of the covariance matrix defined for each point and its neighborhood (see Table I)

As a result, the point cloud C is divided in a set segment ofs S with m components, $S = \{s_1, ..., s_m\}$, where $s_i \subseteq C$. The average values of the previously mentioned attributes in a segment are used to enrich the feature vector in the segmentation sub-network. The attributes length, surface, and segment volume are also concatenated. They are defined in [11], using the eigenvalues $\lambda_{s1} \geqslant \lambda_{s2} \geqslant \lambda_{s3}$ of the covariance matrix of the points into a segment (see Table I). S has dimension F,

(a) Ground-truth (b) Segmentation (c) Our result

Fig. 5: Qualitative results for Forest 3D dataset [34]: (a) Ground-truth with labels for trees, terrain and dead wood. (b) False color representation of the unsupervised segmentation output. (c) The final output for our semantic segmentation pipeline.

which is the feature dimensionality of a given layer. Thus, in the first layer, it corresponds to the input dimension ($F = 3$). Each following layer conducts convolutions on the output of the previous layer.

Consider a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ representing the structure of a segmented point cloud using nodes $\mathcal{V} = \{1, ..., n\}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. In our approach, we construct $\mathcal{G}$ as the k-nearest neighbor (k-NN) graph in $\mathbb{R}^F$. Directed edges are defined as $(i, j_{i1}), ..., (i, j_{ik})$ and $s_{j_{i1}}, ..., s_{j_{ik}}$ are the k closest segments to s_i. Furthermore, we define edge features e_{ij}: $h_\Theta(s_i, s_j)$, where $h_\Theta : \mathbb{R}^F \times \mathbb{R}^F \Rightarrow \mathbb{R}^{F'}$ is a non-linear function parameterized by the set of learnables parameters Θ. In this paper, we set $h_\Theta(s_i, s_j)$ to be an asymmetric edge function of the form $h_\Theta(s_i, s_j) = h_\Theta(s_i, s_j - s_i)$, combining both the global shape structure (captured by s_i) and local neighborhood information (captured by $s_j - s_i$).

The output of edge convolution [10] at node s_i is defined by applying an aggregation operation on the k edge features associated with s_i. In our case, the aggregation operation is defined as maximum value and h_Θ as a mlp:

$$s'_i = \max_{j:\,(i,j)\in\mathcal{E}} h_\Theta(s_i, s_j) \quad . \tag{2}$$

D. Batch Division Strategy

To reduce the impact on the performance of the proposed method, we propose a smart batch division strategy. As mentioned in section III-A, a point cloud initially contains A points. After over-segmentation, we reduce it into B segments. These values are different for every scene, depending mainly on the size, the density, and the number of objects. Table II shows the number of points and segments for the Paris-Lille-3D dataset, which we later use in our experiments, as well. The semantic segmentation sub-network needs to infer the class of the segment S_i the global feature vector from the classification sub-network of S_i and of its k nearest neighbors. Therefore, the minimum batch size bz must be equal to $k+1$ when only one segment is inferred. A batch size, $bz = N \gg k + 1$ depends on the GPU memory size. Consequently, an efficient organization of the input segments increases the number of inferred segments for a batch. This is necessary to reduce the computation time and to use the GPU resources as best as possible. The methodology is explained below.

TABLE III: Quantitative results for the Forest 3D dataset [34]: F1-score is averaged over all three classes. We achieved similar results. Our approach performs slightly better overall and in particular for the class *tree*.

	Precision			**Recall**			**F1**
	tree	terrain	d.w.	tree	terrain	d.w.	avg
HOR [17]	95.0	66.3	**81.8**	97.9	81.2	22.6	69.3
SHOT [17]	93.3	74.6	65.4	98.4	**87.3**	17.4	66.5
3DMI [17]	96.0	**86.2**	66.0	98.7	79.7	**53.1**	79.6
Ours	**96.3**	85.3	74.3	**99.6**	82.5	48.5	**80.1**

First, the k-NN is computed for each segment. Next, an arbitrary initial input segment S_0 and its k-NN, $S_{0_{knn}} = \{S_{0,1}, ..., S_{0,k}\}$ is selected. We define the initial batch subset as $batch_0 = \{S_0 \cup S_{0_{knn}}\}$. Additionally, each neighbor segment $S_{0,j}$ has its own subset of k-NN, $S_{0,j_{knn}} = \{S_{0,j,1}, ..., S_{0,j,k}\}$. Later, in an iterative process until we obtain N elements in the batch subset, we assign as $S_{i+1} \in S_{i_{knn}}$ the segment $S_{i,j}$ with the largest intersection between its subset $S_{i,j_{knn}}$ and the batch subset. The batch subset is updated as $batch_{i+1} = batch_i \cup S_{i,j_{knn}}$.

IV. Experiments

In this section, we compare our results with the state-of-the-art using multiple outdoor LiDAR datasets covering forestal and urban areas. Each series of experiments focuses on a different aspect covering quality improvements, scalability, and the augmentation procedure for the geometric attributes. The performance values of competing approaches mentioned in the following were taken from the literature. We implemented our framework (see Fig. 2) in Python 3.5 using open source libraries and TensorFlow 1.14 as deep learning framework. Training and testing were done using a single Nvidia Tesla v100. During training, we used the ADAM optimizer [36] with initial learning rate $\alpha = 0.001$.

A. Forest Areas

To evaluate our approach in a typical geoscience scenario we used the *3D Forest* dataset [34]. It consists of $467,211$ points recorded using a terrestrial LiDAR scanner. The dataset covers an area containing multiple trees with labels

TABLE IV: Quantitative results for the Forest 3D dataset [34] labeled with four classes using 2-fold cross-validation based on the split proposed in [17]. Both folds perform differently. At the same time, dividing the class tree leads to an overall improvement of performance compared to our result presented in Table III.

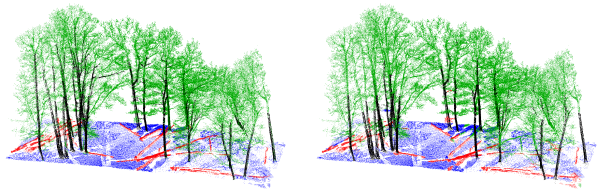
Splits	**Precision**				**Recall**				**F1**
	leaves	terrain	dead wood	trunk	leaves	terrain	dead wood	trunk	avg
Fold 1 – Like [17]	84.2	95.1	86.1	95.1	95.7	96.2	78.7	78.6	88.4
Fold 2 – Reversed	89.6	90.4	81.5	81.6	94.4	86.6	52.6	80.2	81.3
Average	86.9	92.8	83.8	88.3	95.1	91.4	65.7	79.4	84.9

for the semantic classes *tree*, *terrain*, and *dead wood*. An additional *miscelaneous* class exists for partial objects and background scatter. We follow the experimental setup by [17] and remove that class from the evaluation. Furthermore, we adapt their splits for training and testing sets in our initial experiment. A visualization of the complete dataset and its ground-truth labeling can be found in Fig. 5a. After applying unsupervised segmentation, we have segments as visualized using false-color composition in Fig. 5b.

1) Default Setting: For evaluation of semantic segmentation quality, we follow [17] in using the common criteria *precision*, *recall* and *F1-score*. The latter is the harmonic mean of the first two performance measures. Thus, it is a good indicator for the overall best performance. We summarize our results in Table III, where the best performances for each measure are highlighted in bold font, respectively. As can be seen, our approach demonstrates the highest F1-score, as well as the best performance for the most prominent class *tree*. A qualitative comparison between our result and the ground-truth is shown in Fig. 5. In addition to obtaining best performance, our method has a lower computational effort than the method described in [17]. It does not need to compute features for every single point. Although Fig. 5 indicates the point cloud as a flat surface, and it is a hill with a high degree of inclination. We observed that our method is not affected by the terrain surface. We believe this to be the result of our surface normalization process (NDSM).

2) Four Class Setting: A typical task in biological sciences is to measure tree trunks to infer biomass prediction, tree volume, and wood density [37], Thus, we carried out a second set of experiments on the Forest 3D dataset for that task. In particular, we divided the class *tree* manually into two new sub-categories. Those classes are *leaves and tiny branches* and *trunk and significant branches*. Fig. 6a shows the new ground-truth point cloud with four classes. For the evaluation, we applied the same protocol as in the previous experiment. However, we extend it to a two-fold cross-validation strategy. *Fold 1* corresponds to the configuration proposed in [17], which we used in the previous experiment. In contrast, *Fold 2* has reversed splits for training and testing. Thus, we can see how much impact the selected area used for training has.

The results of this experiment can be found in Table IV. Accordingly, a qualitative comparison between our result and the ground-truth is visualized in Fig. 6. As can be seen, our approach exhibits high performance for all classes, including



(a) Ground-truth (b) Our result

Fig. 6: Forest 3D dataset [34] with four labeled classes: (a) Ground-truth with labels for leaves and tiny branches, terrain, dead wood and *trunk and significant branches*. (b) The final output for our semantic segmentation pipeline.

the new two classes. It is important to note that by separating the class tree into two sub-classes, the performance has improved in general. We believe this is due to better defined classes and, thus, less intra-class and more inter-class variance. In the three-class setting, the *tree* class is based on features that describe both leaves and trunks. However, they have completely different geometric characteristics, such that this class spans a wider area in feature space. Please also note that the results of both cross-validation runs are considerably different in their performance. The reversed training strategy resulted in a performance drop of almost 10%. It indicates that performance rates reported in [17] are optimistic.

B. Urban Areas

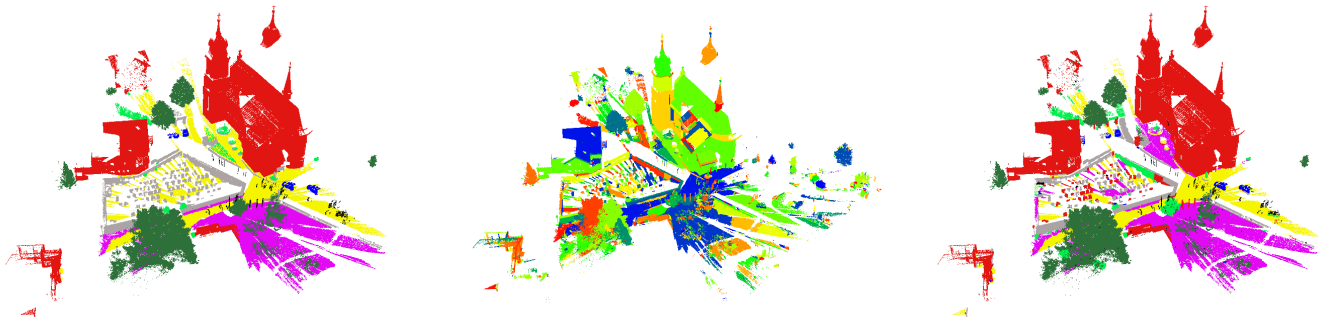
In our second set of experiments, we focus on LiDAR point cloud data recorded in urban areas. For that task, we compare our approach with state-of-the-art in the *Semantic3D* dataset [30] and *Paris-Lille-3D* dataset [35]. In the former, we show the scalability of our approach to large-scale point clouds with millions of 3D points. Afterward, we look into the feature augmentation procedure for our proposed geometric attributes.

Please note, the evaluation of the test data is computed directly at the dataset provider using intersection over union (IoU) [43] and overall accuracy (OA) for Semantic3d and IoU for Paris-Lille-3D as performance measures, respectively.

1) Semantic3D Dataset: Semantic3D [30] consists of 30 labeled urban scenes with a total of over three billion points. Each point is represented by RGB color values and x, y-, z-coordinates for geometric information. The dataset has eight classes covering *human-made terrain*, *natural terrain*, *high vegetation*, *low vegetation*, *building*, *hardscape*, *scanning ar-*

TABLE V: Results on Semantic3D [30] for both the semantic-8 and reduced-8 setting. Values represent intersection-over-union (IoU) scores per class. Classes are *human-made terrain* (C1), *natural terrain* (C2), *high vegetation* (C3), *low vegetation* (C4), *building* (C5), *hardscape* (C6), *scanning artifacts* (C7) and *car* (C8). Additionally, IoU avg. is the averaged intersection-over-union over all the classes and OA is the overall accuracy.

Method	IoU for each class C1	C2	C3	C4	C5	C6	C7	C8	IoU avg.	OA
Semantic-8 Benchmark										
SPGraph [11]	91.5	75.6	**78.3**	**71.7**	**94.4**	**56.8**	**52.9**	**88.4**	**76.2**	**92.9**
PointGCR [25]	**93.8**	**80.0**	64.4	64.4	93.2	39.2	34.3	85.3	69.5	92.1
SnapNet [12]	89.6	79.5	74.8	56.1	90.9	36.5	34.3	77.2	67.4	91.0
PointNet++ [21]	81.9	78.1	64.3	51.7	75.9	36.4	43.7	72.6	63.1	85.7
TMLC-MS [19]	91.1	69.5	32.8	21.6	87.6	25.9	11.3	55.3	49.4	85.0
TML-PC [38]	80.4	66.1	42.3	41.2	64.7	12.4	0.0	5.8	39.1	74.5
Ours	91.1	69.5	65.0	56.0	89.7	30.0	43.8	69.7	64.4	89.6
Reduced-8 Benchmark										
SPGraph [11]	**97.4**	**92.6**	**87.9**	**44.0**	**93.2**	31.0	**63.5**	76.2	**73.2**	**94.0**
MSDeepVoxNet [39]	83.0	67.2	83.8	36.7	92.4	**31.3**	50.0	**78.2**	65.3	88.4
RFMSSF [40]	87.6	80.3	81.8	36.4	92.2	24.1	42.6	56.6	62.7	90.3
SEGCloud [41]	83.9	66.0	86.0	40.5	91.1	30.9	27.5	64.3	61.3	81.1
SnapNet [12]	82.0	77.3	79.7	22.9	91.1	18.4	37.3	64.4	59.1	88.9
3DFCNN-TI [42]	84.0	71.1	77.0	31.8	89.9	27.7	25.2	59.0	58.2	84.8
TMLC-MSR [19]	89.8	74.5	53.7	26.8	88.8	18.9	36.4	44.7	54.2	77.2
TML-PCR [38]	72.6	73.0	48.5	22.4	70.7	5.0	0.0	15.0	38.4	74.0
Ours	84.5	70.9	76.6	26.1	91.4	18.6	56.5	51.4	59.5	87.9



(a) Ground-truth (b) Segmentation (c) Our result

Fig. 7: Qualitative result for one test scene of the Semantic3D dataset [30]: (a) Ground-truth for classes human-made terrain, natural terrain, high vegetation, low vegetation, buildings, hardscape, scanning artifacts, cars. (b) False color representation of the unsupervised segmentation output. (c) The final result for our approach.

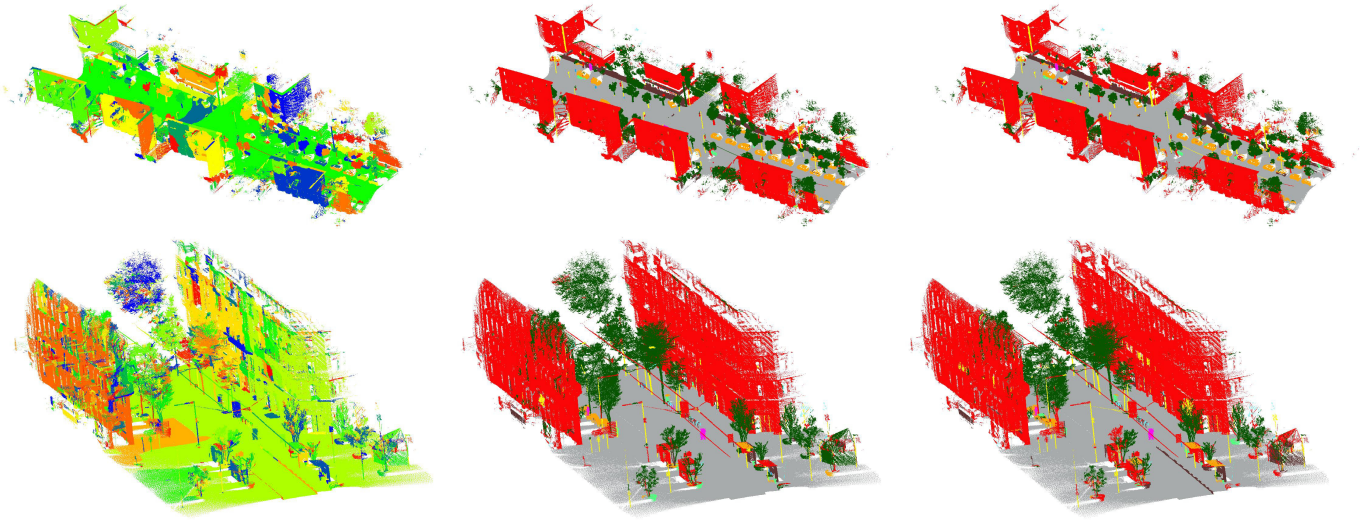
tifacts and *cars*. For benchmark purposes, there exist two settings. The *Semantic-8* benchmark uses complete point clouds with the number of points mentioned above. In comparison, the *Reduced-8* dataset consists of the same training data as the original set. However, the testing set is reduced in size for faster testing [30].

For Semantic-8, we trained our approach using both color and geometric information, while for Reduced-8, we trained using only geometric information. The RGB information or intensity values depend on the recording sensors and are frequently affected by light conditions. Additionally, those values are not always available. Thus, using the Reduce-8 benchmark, we also show how geometric attributes are sufficient to segment point clouds semantically. Our results for Semantic-8 and Reduced-8 setting are summarized in Table V. We report performance for measures *overall accuracy* (OA) and average IoU. Additionally, we show individual performance for the eight classes using their respective IoU. Please note, that OA does not take imbalanced classes into account and is dominated by the performance of most frequently occurring classes. As before, the best results for each class and total performance are highlighted in bold font.

For the Semantic-8 setting in the upper part of Table V, it is visible that classes such as building, human-made terrain, natural terrain, and high vegetation achieve high scores. At the same time, classes such as low vegetation, hardscape, scanning artifacts, and cars achieve low scores, decreasing the overall average performance on the benchmark. In comparison, in the Reduced-8 setting (bottom), classes hardscape and low vegetation show low performance for our approach. There is often a confusion between low vegetation and building structures, mainly when the low vegetation objects are small bushes with low elevation. This confusion of our algorithm between those two classes is

TABLE VI: Quantitative results for the *Paris-Lille-3D* dataset [35], where we compared two different fusion variants. The dataset has labeled classes ground (C1), building (C2), pole (C3), bollard (C4), trash can (C5), barrier (C6), pedestrian (C7), car (C8) and vegetation (C9). Early fusion is preferable for this dataset.

	IoU for each class									IoU
	C1	C2	C3	C4	C5	C6	C7	C8	C9	avg
KP-FCNN [44]	**99.5**	94.0	**71.3**	**83.1**	78.7	47.7	**78.2**	**94.4**	**91.4**	**82.0**
MS3-DVS [39]	99.0	**94.8**	52.4	38.1	36.0	**49.3**	52.6	91.3	88.6	66.9
HDGCN [45]	99.4	93.0	67.7	75.7	25.7	44.7	37.1	81.9	89.6	68.3
RF-MSSF [40]	99.3	88.6	47.8	67.3	2.3	27.1	20.6	74.8	78.8	56.3
Ours (early fusion)	99.3	92.1	37.7	60.7	**78.9**	35.6	28.2	83.2	84.5	66.7
Ours (late fusion)	98.5	84.5	36.7	34.9	27.8	34.1	24.0	68.5	64.6	52.6



(a) Segmentation (b) Early fusion result (c) Late fusion result

Fig. 8: Qualitative results for the *Paris-Lille-3D* dataset [35] for the test point clouds *Ajacio* (top) and *Dijon*: (a) Output of the unsupervised segmentation method in false color composition, (b) our results for early fusion, and (c) our results for late fusion. Colors in results indicate classes ground, buildings, pole, bollard, trash can, barrier, pedestrian, car and vegetation, respectively. In late fusion there is a lot more confusion between small object classes.

also visible in Fig. 7. In general, our approach does not reach state-of-the-art results for this dataset but establishes a competitive performance of almost 90% overall accuracy for the scenes. While classifying segments is more efficient than a point-wise classification of most of the competing works, it can also lead to misclassifications of larger areas. The configuration of the unsupervised segmentation step also has a significant influence on performance. However, we did not optimize this step in our experiments, yet.

2) Paris-Lille-3D: In addition to Semantic3D [30], we evaluated our approach on another urban LiDAR dataset called *Paris-Lille-3D* [35]. It is a benchmark dataset for point cloud classification with dedicated training and testing sets. The training set consists of four scenes with nine labeled classes including *ground*, *building*, *pole*, *bollard*, *trash can*, *barrier*, *pedestrian*, *car* and *vegetation*. In comparison, the test set consists of three scenes. Each point cloud of the dataset has exactly ten million points. There is no RGB information, but reflectance values, which we did not use in our experiments. Our approach was trained exclusively using geometric information based on x-, y-, z-coordinates.

In contrast to our previous experiments, we not only wanted to estimate the best performance but analyze the feature augmentation step. The input for our semantic segmentation sub-network has two sources (see Fig. 1). Those sources are the output of the aggregation function of the classification sub-network and a vector composed of eleven additional attributes. Thus, for this dataset, we tested two different alternatives for the fusion of these features, namely *early fusion* and *late fusion*. In early fusion, the eleven attributes are concatenated directly to the output of the aggregation function (see Fig. 2b). That vector is used as input in our semantic segmentation sub-network. In late fusion, the semantic segmentation sub-network is only trained using the output of the aggregation function of the classification sub-network. Instead, the eleven attributes are concatenated to the last set of mlp-layers (see Fig. 2b).

In Table VI, we give an overview of the results for both analyzed fusion cases. *KP-FCNN* outperforms all methods, and we achieve results in the same range as the remaining ones. Our method using early fusion obtains the better result

TABLE VII: F1-score for the *Paris-Lille-3D* dataset [35], where we compared our method using different input data for the semantic segmentation sub-network. The dataset has labeled classes ground (C1), building (C2), pole (C3), bollard (C4), trash can (C5), barrier (C6), pedestrian (C7), car (C8) and vegetation (C9).

Configuration	**F1 for each class**									**F1**
	C1	C2	C3	C4	C5	C6	C7	C8	C9	avg
all	**86.0**	**78.9**	**51.5**	39.8	**46.2**	28.3	**27.0**	72.3	**61.9**	**54.63**
$d+3f$	82.9	77.6	46.7	**43.9**	38.5	**36.8**	5.1	74.6	58.3	51.79
$d+8f$	71.5	69.6	43.4	17.7	36.2	16.5	0.0	70.0	52.5	41.96
d	69.6	71.8	41.9	0.0	30.7	10.8	0.0	65.6	56.3	38.55
$11f$	84.3	76.3	40.6	0.0	0.0	29.1	0.0	**79.2**	56.9	39.73

for the trash can class. The class *barrier* is often confused with the building and vegetation classes, when the barrier is not well separated from a building or has bushes on top, respectively. We observed that several times the error is originated in the over-segmentation step. Segments contain parts of vegetation mixed with parts of the barrier. The problem could be reduced by changing the unsupervised parameters or selecting another method. It can be seen that our approach performs considerably better using early fusion as opposed to late fusion. The best-classified classes are ground, building, cars, and vegetation that also correspond to the more prominent objects in the scenes. In contrast, the small class pole is often misclassified as a tree trunk (vegetation class). In both fusion setups, the class pedestrian (*C7*) shows the weakest performance. Additionally, barriers are often confused with buildings.

With late fusion, we intended to force the network to focus more on the eleven attributes for the classification decision. In the Paris-Lille-3D dataset, street and sidewalk belong to the same class (ground). As can be seen in Fig. 8(c), there is a small difference in elevation between street and sidewalk, which has the height of a brick. In late fusion, that difference in the height of the segment leads the network to consider it as the class barrier. In contrast, early fusion considers it as a ground class.

Our method aims at being an automatic learning method without applying post-processing algorithms. However, it is evident from Fig. 8(c, top), that for late fusion, the class building presents the largest amount of false positives reflected in small segments. Thus, a possible post-processing option for improvement could be first merged adjacent segments of the same class. Then, segments on the ground with labels building need to be refined. Independent of such an algorithm, we conclude that incorporating basic attributes in the final steps hinder their effectiveness. However, we believe it to be beneficial in cases where prior knowledge indicates that certain geometric attributes are crucial for correct classification.

C. Ablation Study

The semantic segmentation sub-network introduced in section III takes as input the feature vector of dimension d generated for the classification sub-network and additional attributes presented on Table I. In this last experiment, we test five configurations of the input data for the semantic segmentation sub-network. Configuration (all) is our default model, which takes as input the concatenation of the feature vector and the eleven attributes. In comparison, $(d+3f)$ takes as input the feature vector of $d = 256$ and the three features related to the NDSM. For $(d + 8f)$ the feature vector and the attributes are used excluding the three NDSM features. Configuration (d) takes as input only the original feature vector, while $(11f)$ only uses the eleven attributes (see Table I).

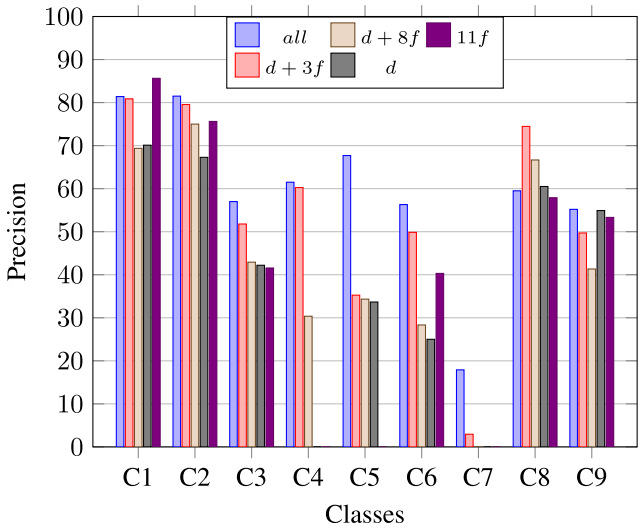


Fig. 9: Precision for the *Paris-Lille-3D* dataset [35], where we compared our method using different input data for the semantic segmentation sub-network. Classes are listed in Table VII. There is a clear advantage of using additional attributes in this study.

We ran this experiment on the *Paris-Lille-3D* dataset [35]. However, the testing part is not publicly available. Therefore, exclusively for this experiment, we divided the official training dataset into a training and validation part. For training we used the scenes Lille 1-1, Lille 2, and Paris, while for validation we used the scene Lille 1-2. During training, we combined ADAM optimizer [36] as mentioned above with an early stopping strategy to avoid overfitting. We did not apply any data augmentation technique.

Table VII shows the F1-score for all nine classes and the average F1 for the validation segments. We can observe that the (all) configuration presented the highest precision with exception to the pedestrian class, which is particularly hard to classify in our approach. Fig. 9 shows the precision for each of the classes for the five configurations. The classes

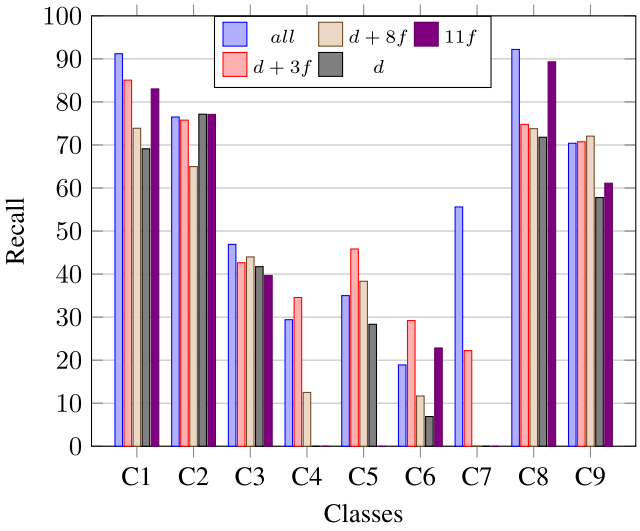


Fig. 10: Recall for the *Paris-Lille-3D* dataset [35], where we compared our method using different input data for the semantic segmentation sub-network. Classes are listed in Table VII. The recall results also show, that additional attributes improve classification performance in comparison to only using the original feature vector (d).

bollard and trash present high precision but lower recall (see Fig. 10). These classes correspond to the objects with smaller size and less presence in the dataset. The (all) model presents the best and most uniform results for both precision and recall measures. The architecture of our network does not require a large context during inference. However, it uses additional features to overcome this limitation. By using more information as input, the network can distinguish the objects with small sizes such as bollards, trash cans, and pedestrians. In summary, we can clearly observe the advantageous effect of using additional information in this study.

V. Conclusions

In this paper, we showed how edge convolution could be adopted for segment-wise semantic segmentation. We proposed a deep learning pipeline, where regions are classified instead of individual points, and geometric attributes are used as additional features. Furthermore, we demonstrated the inclusion of normalized elevation information. It helped to distinguish between objects of low inter-class variances, such as trees and small bushes based on their relative height above ground. In our experiments, we analyzed quality, scalability, and feature augmentation procedure.

For a forest LiDAR dataset, we established a new top performance with 80.1% F1-score. Especially the segmentation of individual trees with precision and recall of 96.3% and 99.6, respectively, is notable. Additionally, we were able to achieve competitive performances on popular urban 3D point cloud benchmarks. On Semantic3D dataset, we obtained an OA of 89.6% and average IoU of 64.4% For the urban dataset Paris-Lille-3D, we reached 66.7% mIoU. Our results on the feature augmentation process indicate that a fusion before edge convolution is preferable when compared to adding information directly in front of the classification layer.

References

[1] H.-E. Andersen, R. J. McGaughey, and S. E. Reutebuch, "Estimating forest canopy fuel parameters using lidar data," *Remote Sens. Environ.*, vol. 94, no. 4, pp. 441–449, 2005.

[2] J. Yang, Z. Kang, S. Cheng, Z. Yang, and P. H. Akwensi, "An individual tree segmentation method based on watershed algorithm and three-dimensional spatial distribution analysis from airborne lidar point clouds," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 13, pp. 1055–1067, 2020.

[3] Z. Abdeldayem, "Automatic weighted splines filter (AWSF): A new algorithm for extracting terrain measurements from raw lidar point clouds," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 13, pp. 60–71, 2020.

[4] Y. Xie, J. Tian, and X. X. Zhu, "A review of point cloud semantic segmentation," *arXiv preprint arXiv:1908.08854*, 2019.

[5] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3d point clouds: A survey," *arXiv preprint arXiv:1912.12033*, 2019.

[6] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*, 2018, pp. 4490–4499.

[7] G. Riegler, A. Osman Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*, 2017, pp. 3577–3586.

[8] X. Xu, J. Amaro, S. Caulfield, G. Falcao, and D. Moloney, "Classify 3d voxel based point-cloud using convolutional neural network on a neural compute stick," in *Int. Conf. Natural Computation, Fuzzy Systems and Knowledge Discovery*. IEEE, July 2017, pp. 37–43.

[9] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*. IEEE, 2017, pp. 77–85.

[10] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, pp. 1–12, 2019.

[11] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*, 2018, pp. 4558–4567.

[12] A. Boulch, B. Le Saux, and N. Audebert, "Unstructured point cloud semantic labeling using deep segmentation networks," in *Eurographics Workshop on 3D Object Retrieval*, 2017.

[13] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud," in *IEEE Int. Conf. ICRA*. IEEE, 2018, pp. 1887–1893.

[14] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *IEEE Int. Conf. Comput. Vis.* IEEE, 2015, pp. 945–953.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards teal-time object detection with region proposal networks," in *Adv Neural Inf Process Syst*, 2015, pp. 91–99.

[16] D. J. Meagher, *Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer*. Electrical and Systems Engineering Department Rensseiaer Polytechnic Institute Image Processing Laboratory, 1980.

[17] S. Sickert and J. Denzler, "Semantic segmentation of outdoor areas using 3d moment invariants and contextual cues," in *German Conf Pattern Recognit.*, ser. Lecture Notes in Computer Science, vol. 10496. Springer, 2017, pp. 165–176.

[18] S. Salti, F. Tombari, and L. D. Stefano, "Shot: Unique signature of histograms for surface and texture description," *Comput. Vision Image Understanding*, vol. 125, pp. 251–264, August 2014.

[19] T. Hackel, J. D. Wegner, and K. Schindler, "Fast semantic segmentation of 3d point clouds with strongly varying density," *Ann. ISPRS*, vol. 3, no. 3, 2016.

[20] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Adv Neural Inf Process Syst*, 2015, pp. 2017–2025.

[21] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Adv Neural Inf Process Syst*, vol. 30. Curran Associates, Inc., 2017, pp. 5099–5108.

[22] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *Adv Neural Inf Process Syst*, 2018, pp. 820–830.

[23] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "Pointsift: A sift-like network module for 3d point cloud semantic segmentation," *arXiv preprint arXiv:1807.00652*, 2018.
[24] L. Ma, Y. Li, J. Li, W. Tan, Y. Yu, and M. A. Chapman, "Multi-scale point-wise convolutional neural networks for 3d object segmentation from lidar point clouds in large-scale environments," *IEEE Trans. Intell. Transp. Syst.*, 2019.
[25] Y. Ma, Y. Guo, H. Liu, Y. Lei, and G. Wen, "Global context reasoning for semantic segmentation of 3d point clouds," in *IEEE Winter Conf Appl. Comput. Vis*, 2020, pp. 2931–2940.
[26] X. Liang and Z. Fu, "Mhnet: Multiscale hierarchical network for 3d point cloud semantic segmentation," *IEEE Access*, vol. 7, pp. 173 999–174 012, 2019.
[27] Y. Li, L. Ma, Z. Zhong, D. Cao, and J. Li, "Tgnet: Geometric graph cnn on 3-d point cloud segmentation," *IEEE Trans. Geosci. Remote Sens.*, 2019.
[28] X. Ye, J. Li, H. Huang, L. Du, and X. Zhang, "3d recurrent neural networks with context fusion for point cloud semantic segmentation," in *Proc. ECCV*, 2018, pp. 403–417.
[29] Y. Xu, Z. Ye, W. Yao, R. Huang, X. Tong, L. Hoegner, and U. Stilla, "Classification of lidar point clouds using supervoxel-based detrended feature and perception-weighted graphical model," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 13, pp. 72–88, 2020.
[30] T. Hackel, N. Savinov, L. Ladickỳ, J. D. Wegner, K. Schindler, and M. Pollefeys, "Semantic3d.net: A new large-scale point cloud classification benchmark," *Ann. ISPRS*, vol. IV-1-W1, pp. 91–98, 2017.
[31] S. Guinard and L. Landrieu, "Weakly supervised segmentation-aided classification of urban scenes from 3d lidar point clouds," *Int Arch. ISPRS*, vol. XLII-1/W1, pp. 151–157, 2017.
[32] K. Zhang, S.-C. Chen, D. Whitman, M.-L. Shyu, J. Yan, and C. Zhang, "A progressive morphological filter for removing nonground measurements from airborne lidar data," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 4, pp. 872–882, 2003.
[33] W. Zhang, J. Qi, P. Wan, H. Wang, D. Xie, X. Wang, and G. Yan, "An easy-to-use airborne lidar data filtering method based on cloth simulation," *Remote Sensing*, vol. 8, no. 6, p. 501, 2016.
[34] J. Trochta, , M. Krůček, T. Vrška, and K. Král, "3d forest: An application for descriptions of three-dimensional forest structures using terrestrial lidar," *PLoS One*, vol. 12, no. 5, p. e0176871, May 2017.
[35] X. Roynard, J.-E. Deschaud, and F. Goulette, "Paris-lille-3d: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification," *Int J Rob Res*, vol. 37, no. 6, pp. 545–557, 2018.
[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[37] J. Hackenberg, M. Wasserberg, H. Spiecker, and D. Sun, "Non destructive method for biomass prediction combining tls derived tree volume and wood density," *Forests*, vol. 6, no. 4, pp. 1274–1300, April 2015.
[38] J. A. Montoya-Zegarra, J. D. Wegner, L. Ladickỳ, and K. Schindler, "Mind the gap: Modeling local and global context in (road) networks," in *German Conf Pattern Recognit.*, ser. Lecture Notes in Computer Science. Springer, 2014, pp. 212–223.
[39] X. Roynard, J.-E. Deschaud, and F. Goulette, "Classification of point cloud scenes with multiscale voxel deep network," *arXiv preprint arXiv:1804.03583*, 2018.
[40] H. Thomas, F. Goulette, J.-E. Deschaud, and B. Marcotegui, "Semantic classification of 3d point clouds with multiscale spherical neighborhoods," in *2018 Int. Conf. 3D Vision*. IEEE, 2018, pp. 390–398.
[41] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," in *Int Conf 3D Vision*. IEEE, 2017, pp. 537–547.
[42] F. J. Lawin, M. Danelljan, P. Tosteberg, G. Bhat, F. S. Khan, and M. Felsberg, "Deep projective 3d semantic segmentation," in *Int Conf Comput Anal Ima Pattern*. Springer, 2017, pp. 95–107.
[43] J. Xu, Y. Ma, S. He, and J. Zhu, "3d-giou: 3d generalized intersection over union for object detection in point cloud," *Sensors*, vol. 19, no. 19, p. 4093, 2019.
[44] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proc IEEE Int Conf Comput Vis*, 2019, pp. 6411–6420.
[45] Z. Liang, M. Yang, L. Deng, C. Wang, and B. Wang, "Hierarchical depthwise graph convolutional neural network for 3d semantic segmentation of point clouds," in *2019 Int. Conf. ICRA*. IEEE, 2019, pp. 8152–8158.

Jhonatan Contreras earned the B.Sc. in Electronic Engineering in 2013 at University Industrial of Santander, Colombia and M.Sc. in Electrical Engineering in 2016 at Pontifical Catholic University of Rio de Janeiro. Brazil. He is currently working towards the PhD degree at the Computer Vision Group of Joachim Denzler at the Friedrich Schiller University Jena. He is also member of the Citizen Science Lab at DLR Institute of Data Science, Jena. His research interests are in deep learning, semantic segmentation and probabilistic graphical models.

Sven Sickert earned the Diploma degree in Computer Science in 2012 from the Friedrich Schiller University Jena, Germany. He received his PhD in 2018 for his work on 3D semantic segmentation using context features, which was done under supervision of Joachim Denzler. He is currently senior researcher and lecturer at the Computer Vision Group Jena. His research interests are in the field of machine learning, local features and analysis of three-dimensional data with a focus on learning using unstructured data.

Joachim Denzler earned the degrees "Diplom-Informatiker", "Dr.-Ing." and "Habilitation" from the University of Erlangen, Germany, in years 1992, 1997, and 2003, respectively. Currently, he holds a position as full professor for computer science and is head of the Computer Vision Group at the Friedrich Schiller University Jena, Germany. His research interests comprise the automatic analysis, fusion, and understanding of sensor data, especially development of methods for visual recognition tasks and dynamic scene analysis. He contributed in the area of active vision, 3D reconstruction, and object recognition and tracking. He is author and co-author of over 300 journal and conference papers and technical articles. He is a member of IEEE, IEEE computer society, DAGM, and GI.