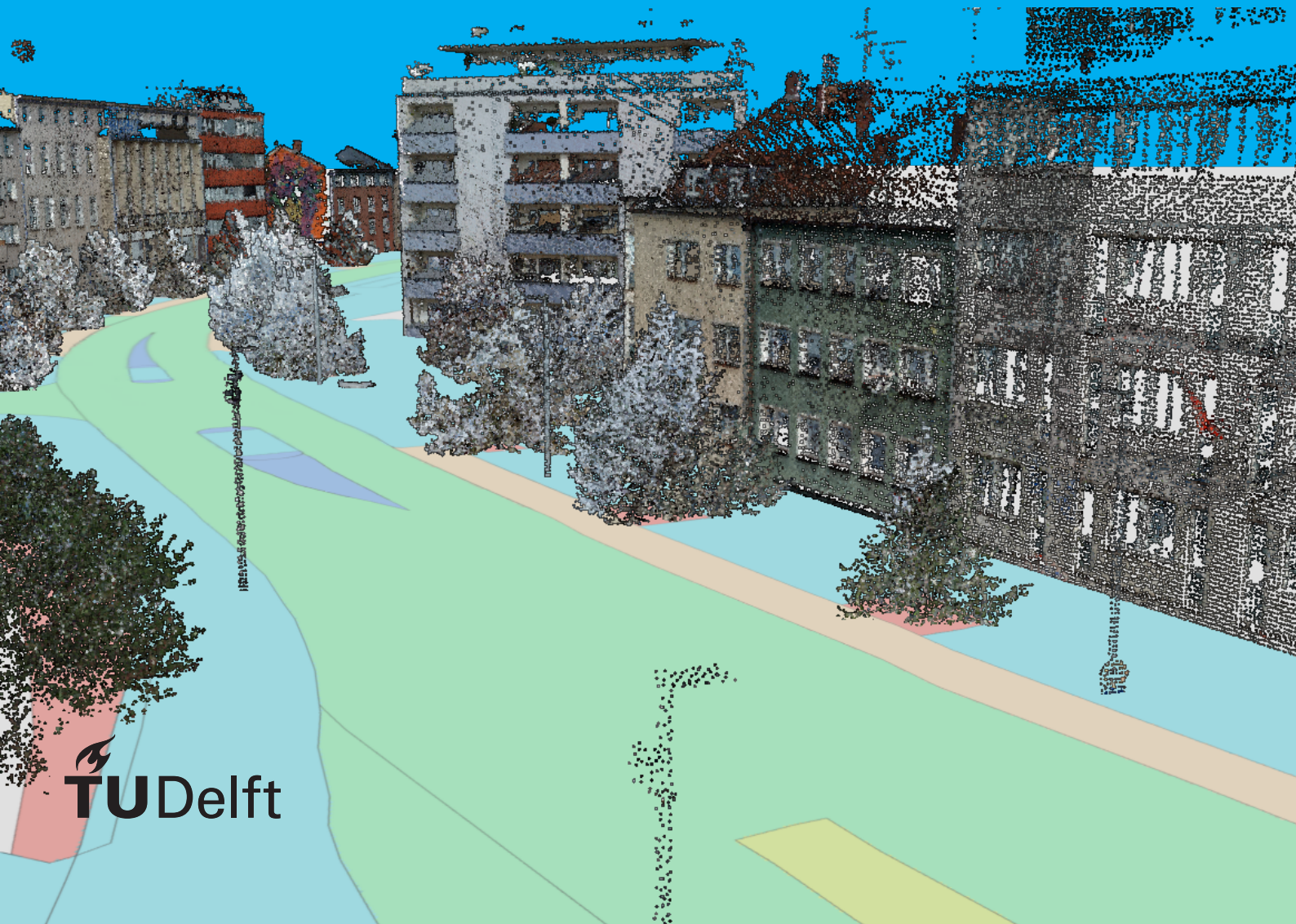
3D Road Boundary Mapping of MLS Point Clouds

Qian Bai

Geoscience & Remote Sensing
Delft University of Technology



TU Delft

3D Road Boundary Mapping of MLS Point Clouds

by

Qian Bai

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on July 8th, 2021.

Student number:	5098750	
Project duration:	October 5, 2020 – July 1, 2021	
Thesis committee:	Dr. R. C. Lindenbergh,	Optical and Laser Remote Sensing group, TU Delft, chair
	Dr. Liangliang Nan,	3D Geoinformation group, TU Delft
	Dr. Riccardo Taormina,	Sanitary Engineering section, TU Delft
	Julien Vijverberg,	Cyclomedia Technology

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Abstract

Roads in modern cities facilitate different types of users, including car drivers, cyclists, and pedestrians. These different users often have a designated section of the road to operate on. Road management, e.g., by municipalities, needs to take this sectioning into account, preferably in an efficient way. Mobile laser scanning (MLS) point clouds provide accurate and dense three-dimensional (3D) measurements of road scenes, showing strong mapping capabilities, although their massive data volume and lack of structure still bring difficulties in automatic processing. Methods for the automatic classification of road surface types are still largely lacking, and the existing methodology did not consider the potential of MLS point clouds yet.

In recent years, point cloud understanding through deep neural networks has achieved breakthroughs. However, perceiving large-scale point clouds by deep learning depends on aggregating local features and progressive downsampling, to extract rich contextual information. As a consequence, low-level features that reveal details in point clouds may not be well preserved, possibly resulting in ambiguous delineation in point cloud classification. For road mapping, inaccurate classification of points on road boundaries hinders the generation of high-quality map products. Some existing deep learning methods propose to mitigate the fuzzy classification near boundaries, either by utilizing refinement for network predictions, or by indirectly modifying neighboring weights when summarizing local information. Approaches to achieving a satisfying overall performance, while maintaining accurate delineation, still need investigation.

In this study, we propose a novel approach for road type classification of MLS point clouds in dense urban areas based on a deep neural network. We follow the main architecture of RandLA-Net, a point-wise neural network designed for large-scale point cloud processing. To alleviate the ambiguous delineation of point cloud classification, we propose two strategies. The first is to refine predictions of RandLA-Net by conditional random field. In the second strategy, we incorporate boundary constraints in the network by introducing a novel distance label for each road surface point to represent the distance to its closest boundary. The distance prediction task is combined with road type classification by adding another branch in RandLA-Net to formulate multi-task learning. Through experiments, we show that 3D point cloud semantic segmentation by deep learning is applicable for road type classification. Also, the multi-task learning strategy is verified to be more effective in improving the delineation performance. Using MLS point clouds acquired from 5 German cities (Hamburg, Delmenhorst, Bremerhaven, Hannover, and Oldenburg), we classify road points separately into different usages (*sidewalk*, *cycling path*, *rail track*, *parking area*, *motorway*, *green area*, and *island without traffic*) and materials (*cobblestone*, *asphalt*, *plates*, *unpaved*, and *railway*). When adopting Hannover and Oldenburg for testing, and the other three cities for training, we obtain a mean intersection over union of 46.1% for usage type and 52.0% for material type with the multi-task learning strategy and input features (x, y, z, R, G, B, intensity), outperforming the original RandLA-Net by approximately 4%. Moreover, from the point cloud classification results, we achieve lightweight polygon representations of road objects in different types through post-processing, which is demonstrated to perform better than an image semantic segmentation-based solution quantitatively and qualitatively.

Acknowledgement

My utmost thanks go to my daily supervisor Roderik, for his guidance, good ideas, and a great deal of encouragement during the research. I am very grateful to Liangliang, who provided valuable suggestions and feedback for both the scientific approach and report writing. I would also like to thank Riccardo for his insights into deep learning methods and comments regarding my thesis.

I would like to express my sincere gratitude to Julien and Jeroen from Cyclomedia. They provided much guidance and support during the last eight months. Without the weekly meetings with them, I would not finish my thesis smoothly during the corona time. I am also grateful to Bas for his important input to this thesis. Special thanks go to Niels from ESRI, who introduced me to the wonderful opportunity of doing an internship in Cyclomedia.

Many thanks to Zhaiyu, who not only helped me with the format and typos of the thesis, but also made my life abroad more enjoyable. Last but not least, I especially thank my parents, for their love and continuous support, and other friends who made sure I had a lot of fun except for the studies.

Qian Bai
Delft, July 2021

Contents

1 Introduction

1.1. Road Boundary Mapping

Roads play a significant role in economic and social development, by connecting different communities and ensuring safe and efficient transportation of people and goods [13]. In modern life, a well-designed road system is classified into detailed functions (e.g., highway, residential street, and bicycle path) in order to appropriately plan each type of facilities. Each road type is determined according to its practical usage or priority for access. Road boundaries, an important road feature, provide delineation of a road object and indicate its accurate position. Up-to-date and reliable road boundary information is key to various applications, including inspection of infrastructures, acquisition of high-definition maps for autonomous driving, and decision making of companies and governments.

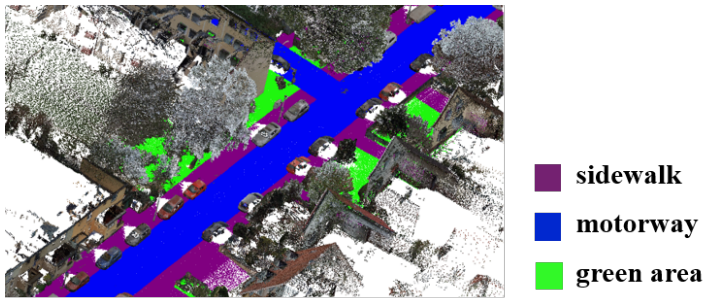


Figure 1.1: Different road functions in urban areas.

To document and update road boundaries, vector formats such as polygons and polylines are widely used as representations on digital road maps. Besides representing the location and shape of a road segment, vectorized boundaries can also hold object attributes, e.g., areas and a semantic label. In addition, they are lightweight, making the storage and modification convenient in practice. However, creating such boundary vectors traditionally depends on massive manual annotations from spatially referenced images, which is labor-intensive and time-consuming, especially for complex urban road environments nowadays. Therefore, timely acquisition and evaluation of road boundary maps are often lacking.

With the rapid development of machine learning, road boundary mapping has also benefited from the automatic understanding of images and three-dimensional (3D) point clouds. For example, object detection techniques based on deep neural networks help to identify curbs along the road. Then road boundaries can be extracted by connecting these curbs. However, this method cannot be directly applied to road boundary mapping with detailed functional divisions in dense urban areas (as shown in Figure 1.1), since not all types of roads have curbs or an apparent height difference between them. Besides curb detection, semantic segmentation using deep learning can help to distinguish different road objects in images or point clouds. Despite this, it is challenging to achieve accurate boundaries from pixel or point labeling by most deep neural

networks due to loss of low-level details through progressive downsampling. Hence, there is still a need for automatic road boundary mapping solutions that can provide accurate delineation of road objects of various types.

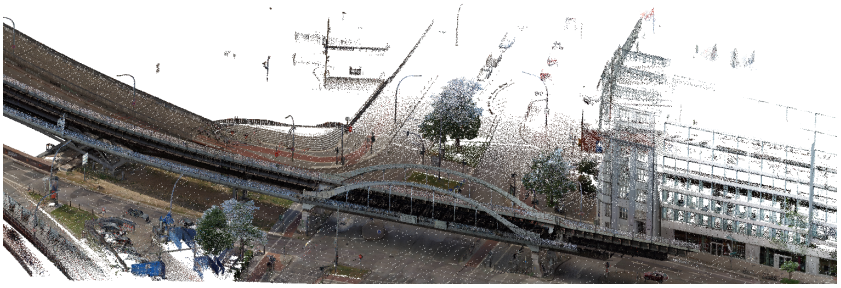


Figure 1.2: An urban point cloud rendered with colors.

1.2. MLS Point Clouds

A point cloud is a set of points in space, as shown in Figure 1.2. Point clouds defined in a 3D coordinate system represent the shape of objects and are widely used to create 3D models (e.g., meshes) for visualization and animation. Considering their ability to describe surface properties, point clouds are also adopted in applications like infrastructure inspection. In general, the acquisition of point clouds can be divided into two categories, i.e., from multi-view imagery and Light Detection And Ranging (LiDAR). In photogrammetry and computer vision, sparse 3D points are often reconstructed from images that contain a series of matched key features by techniques like Structure from Motion [33]. Dense point clouds can be generated afterwards according to multi-view stereo. Besides, a laser scanner, which collects reflections of laser beams that bounce back from object surfaces, directly produces high-resolution 3D point clouds. Combined with Global Navigation Satellite Systems (GNSS) and Inertial Measurement Units (IMU), laser scanners can accurately measure 3D objects.

LiDAR point clouds provide accurate 3D measurements that are illumination invariant, showing a strong ability for mapping. Moreover, the intensity attribute of each point acquired by laser scanners measures the return strength of the laser pulse, which is relative in value and reveals surface properties. Intensity can offer important information for feature extraction in complex scene understanding.

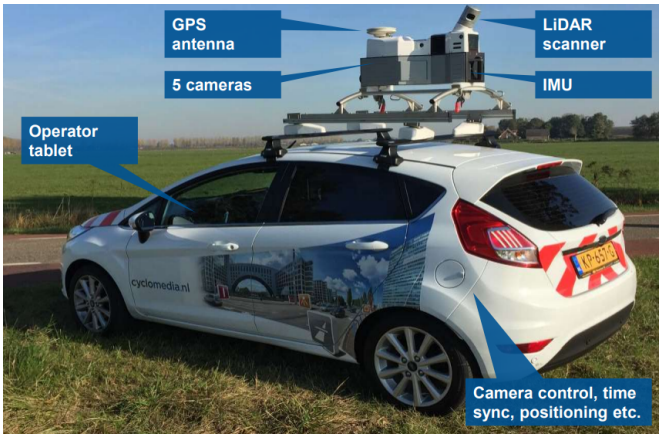


Figure 1.3: DCR10L mobile mapping system from Cyclomedia Technology[1]. D: Digital, C: Cyclorama (Panoramic Image), R: Recording system (version 10), L: Light Detection and Ranging (LiDAR).

For applications that require measurements from extensive road scenes, laser scanners are usually mounted onto mobile platforms like cars. The scanner can then gather massive 3D points along the vehicle trajectory

[1] Image source: https://www.cyclomedia.com/us/product/data-capture/data-capture

in a short period of time. Such data are called mobile laser scanning (MLS) point clouds, which describe the road-related objects including road surfaces, traffic signs, and buildings at the roadside. MLS point clouds hold detailed information of road scenes. Interpretation of MLS point clouds (e.g., semantic segmentation) is showing great potential for producing large-scale and high-definition road maps. In practice, laser scanners together with GNSS, cameras, or any other remote sensing devices, constitute a multi-functional mobile mapping system, as shown in Figure 1.3. Geo-referenced images acquired at the same time can also help to render MLS point clouds with colors.

1.3. Deep Learning for Point Cloud Understanding

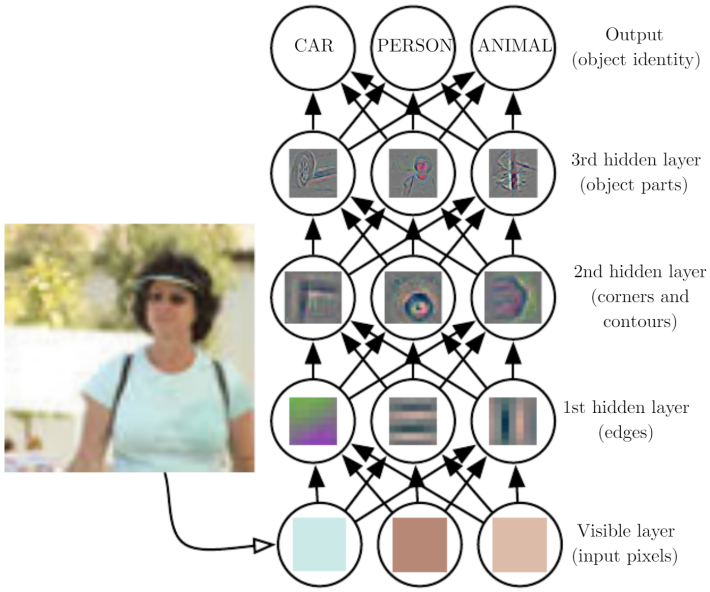


Figure 1.4: An example of a deep learning algorithm. In deep learning, a person in this image is recognized from simpler representations such as corners and edges [10].

Deep learning, as a branch of machine learning, enables computers to learn from data by building complicated concepts on top of simpler ones [10]. The idea imitates the thinking process of humans and is realized by layers of models. Each layer extracts a certain level of information from input features that are the output of previous layers (see Figure 1.4). The weights of each layer are optimized during the training process. In 1979, Kunihiko Fukushima developed the first convolutional neural network (CNN), allowing the computer to understand visual patterns through convolutional and pooling layers [9]. In recent years, deep learning has been vastly used in applications that are closely related to human life like speech understanding and image recognition.

Although deep learning has promoted huge progress in understanding two-dimensional (2D) images, processing 3D data (e.g., point clouds) with neural networks is relatively immature. Unlike images, in which pixels are regularly aligned, points in a point cloud are unordered, resulting in difficulties for automatic processing. Such unorganized data format cannot provide explicit neighborhood relations as in images, making conventional CNNs infeasible for point cloud understanding [11]. In addition, the sheer data volume of some point cloud data, such as MLS point clouds, can hamper the processing efficiency. Hence, to accomplish automatic understanding of 3D point cloud, e.g., retrieving labels of each point in semantic segmentation, one approach is to project the 3D data onto image planes to make use of state-of-the-art 2D CNNs. This method avoids directly processing 3D data and reduces the data volume, but also causes loss of geometric information to some extent. Point-wise neural networks like PointNet [28] have led to breakthroughs in 3D point cloud semantic segmentation with deep learning techniques, which facilitate the end-to-end processing of point clouds. A detailed description of deep learning methods for point cloud semantic segmentation will be presented in Chapter 2.2.

1.4. Problem Statement and Our Approach

MLS data has a good coverage of road scenes and can depict various types of roads with high resolution, which is beneficial to elaborate road mapping. In this thesis project, we aim to achieve vectorized road boundaries through road type classification of MLS point clouds. Specifically, we focus on road scenes in dense urban areas and desire an automatic solution for classifying MLS point clouds on a large scale. The performance of point cloud classification[2], especially delineation performance, is crucial to the following vector-based road boundary generation.

The type of a road segment in urban areas can be determined according to its functionality or surface material, depending on the practical requirements. Both types play an important role in the sustainable development of modern cities. In terms of functionality, it happens that a road object is used for multiple purposes, e.g., bicycle lane and motorway. The labeling of such a road segment can rely on a priority list. For example, it should be classified as bicycle lane instead of motorway if green transport is more valued by the government and citizens. The need for detailed division of urban roads and additional rules increase the complexity of urban road scenes, bringing challenges into automatic road type classification. Therefore, we need an automatic method that is able to extract prominent features from large-scale MLS point clouds to perceive complex scenes.

To facilitate road boundary mapping, a focus should also be put on the accurate delineation in point cloud classification results. The semantic label of each point cannot be acquired by looking at an individual point, making it necessary to exploit point neighborhood relations. In areas near object boundaries, constructing local features in a neighborhood (e.g., K-Nearest Neighbors) might aggregate features from points belonging to different types of objects. Also, a progressive downsampling strategy is commonly used in existing deep learning methods for point cloud classification to achieve large receptive fields and high-level semantics, which drops low-level but high-resolution information to some extent. As a result, fuzzy boundaries can be observed in the classified point clouds. With manual annotation, confusion near boundaries can be corrected by setting hard constraints (e.g., a height threshold), which is not feasible for deep learning approaches.

In this study, we propose two strategies to improve the delineation performance in road type classification of MLS point clouds based on an existing point-wise neural network. The first method is to refine the predictions of the original network by encouraging points with similar features to have the same label using Conditional Random Field (CRF). In the second strategy, inspired by the distance transform in images, we explicitly incorporate boundary information in network training by encoding the distance from each point to its closest road boundary as another label and implementing multi-task learning. The optimization of distance predictions during training can be regarded as a constraint for the road type classification task. Through extensive evaluation, we demonstrate that the second method is much more effective and shows more robust performance in the case of complex scenes where different types of road points have relatively indiscriminate features.

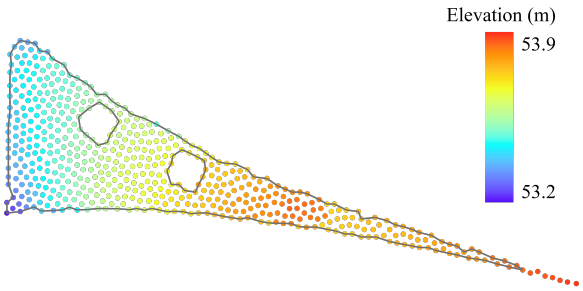


Figure 1.5: 2D polygon representing the shape formed by a set of points belonging to a traffic island, with colors showing the elevation of points.

Having the road type classification results of MLS point clouds, we also generate road boundary vectors by fitting a polygon around each classified road segment. As illustrated in Figure 1.5, the polygon should be closely associated with the shape described by a type of road points. In the end, we deliver the vectorized

[2] In this thesis, both terms “point cloud classification” and “point cloud semantic segmentation” refer to the per-point labeling task and will be used interchangeably.

road boundaries in a common vector data storage format (i.e., ESRI Shapefile), which documents both the precise position and road type of the boundaries.

1.5. Research Questions

The main research question for this project is summarized as:

- ***How to acquire road types from MLS point clouds accurately and efficiently?***

The thesis is further split into the following sub-questions:

1. What kind of pre-processing strategies should be applied to MLS point clouds?
2. How to realize road type classification of MLS point clouds through deep learning?
3. How to alleviate the fuzzy boundary issue in 3D point cloud classification?
4. How is the generalization ability of the proposed method regarding different point cloud datasets?
5. Given the road type classification results, how to extract the road boundary vectors effectively?
6. How good are the road boundaries achieved by our method compared to other methods, e.g., image-based approaches?

1.6. Thesis Structure

The rest of this thesis is organized as follows:

Chapter 2 presents previous studies related to road boundary mapping and point cloud semantic segmentation using deep learning. Chapter 3 describes the adopted methodology, including data pre-processing, road type classification of MLS point clouds as well as delineation improvement, road boundary vector extraction through post-processing, and evaluation metrics. In Chapter 4, we introduce the predefined road types and the MLS point cloud dataset used in this project. Afterwards, Chapter 5 shows the results of road type classification. Chapter 6 further discusses the quality of extracted road boundary polygons and the impact of different point densities on road type classification. Finally, Chapter 7 concludes the thesis by answering all research questions. Several recommendations for future research are also provided.

2

Related Work

This chapter firstly introduces methods for extracting road information automatically in Section 2.1. Section 2.2 presents recent research on point cloud semantic segmentation using deep learning. Moreover, Section 2.3 discusses the ambiguous delineation problem in semantic segmentation tasks and existing solutions.

2.1. Automatic Road Information Extraction

Road infrastructures consist of both road surfaces and their surrounding environment including road markings, traffic signs, power lines, vegetation, bridges, and buildings, etc [13]. Extracting road information (e.g., road surface type, center lines, and boundary lines) from surveyed data such as images and Light Detection and Ranging (LiDAR) point clouds often depends on a comprehensive understanding of the whole road scene. This information is necessary for forming an up-to-date road inventory database.

Figure 2.1: Roads detected from Planet satellite imagery[1].

Early studies of road scene understanding mainly focused on raster images. Satellite and aerial images are applied in road network surveying due to their wide coverage. From these images, it is possible to extract road features like center lines which represent the geographic center of roads [19]. However, although a good overview of the road network can be provided, processing large-scale images can suffer from a lack of precision sometimes, as a result of balancing the coverage and granularity [34]. Also, precise road geometry cannot always be retrieved from large-scale images with relatively low resolution [17]. By contrast, street-view images represent more details of road scenes. Rateke et al. classified road surfaces into asphalt, paved, and unpaved

[1] Image source: https://www.planet.com/pulse/crowdai-webinar/

road from images captured by a webcam using CNNs [31]. However, the performance on street-view images can still be affected by ambient illumination conditions. For instance, shadows and bright sunlight during image acquisition might cause a false interpretation of road scenes.

Compared to cameras, LiDAR scanners produce robust and accurate 3D point measurements that are not influenced by illumination changes. Airborne laser scanning is able to acquire road information on a large scale and is used to extract road networks (e.g., center lines) for mapping [43]. In recent years, LiDAR point clouds have become more and more accessible with the development of mobile laser scanning technologies. Vehicles with an MLS system can record data along different types of roads (e.g., highway and rural road), capturing extensive road objects and surroundings. Mobile point cloud data is often acquired with a high resolution and precise geo-locations if coupled with GNSS systems, making them reliable for the inventory creation and inspection of road infrastructures. In order to provide interpreted road scenes, most algorithms rely on exploiting neighborhood relations inside the point cloud to extract the road geometry. Such relations are usually modeled based on a neighborhood construction method (e.g., K-Nearest Neighbors) and a way of calculating prominent features from the local region (e.g., principle component analysis). Detection of objects, such as curbs and planar surfaces, benefits a lot from their unique geometric shapes. Balado et al. automatically segmented ground in the urban areas based on a geometric decision tree and adjacent constraints [2]. Moreover, LiDAR features, e.g., reflectance intensity, play a vital role in distinguishing objects like road markings and vegetation areas due to their special surface properties. Widyaningrum et al. took advantage of the LiDAR intensity feature for the classification of aerial urban point clouds [44].

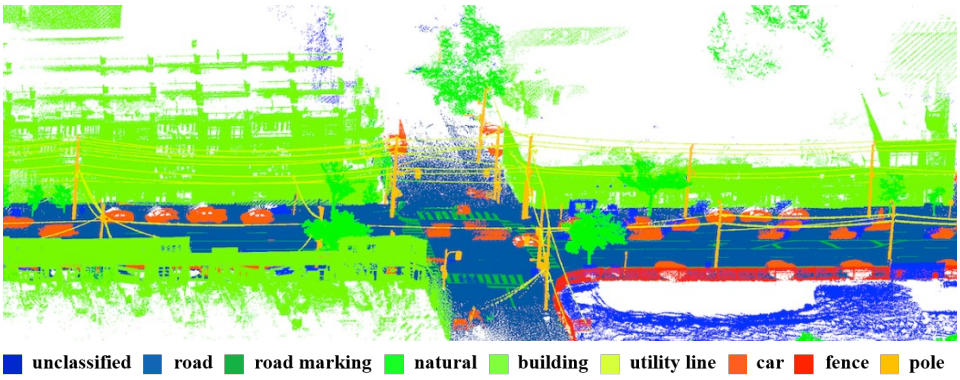


Figure 2.2: Point cloud semantic segmentation of the Toronto-3D dataset [37].

Semantic information of the road environment forms a basis for further processing and analysis of MLS point clouds (see Figure 2.2). Semantic segmentation aims to partition the original point clouds into different types, achieving smaller data volume and lower scene complexity in each subset. As a common strategy for semantic segmentation of both images and point clouds, region growing begins with seed points and iteratively adds the neighboring points that have similar properties (e.g., orientation) to obtain homogeneity within regions. Vo et al. proposed an octree-based region growing approach to realize fast surface patch segmentation of 3D urban point cloud in a coarse-to-fine manner [40]. However, region growing methods are sensitive to the choice of initial seed points. They can also be influenced by the inaccurate estimation of geometric features (e.g., normal and curvature) near region boundaries [12]. Semantic segmentation based on model fitting is also applied to many man-made objects that are composed of geometric primitives , e.g., cylinders and spheres. Hough Transform [3] and Random Sample Consensus (RANSAC) [8] are widely used for fitting these simple shapes. Although they are fast and robust, model fitting methods are not feasible for objects with complex shapes. Nowadays, semantic segmentation of MLS point clouds benefits from deep learning techniques as well. Soilán et al. classified the railway tunnel into ground, lining, rails, and wiring from the MLS point cloud with PointNet and KPConv networks [35]. Deep learning methods avoid constructing handcrafted geometric features using principle component analysis (PCA). However, a large amount of ground truth data is required.

In regard to **road boundary extraction** using MLS point clouds, there are mainly two directions to achieve automatic solutions. First, road boundary extraction of point cloud data can be related to the detection of common patterns (e.g., curbs) along the road. In some cases, the elevation difference between curbs and the

road surface is beneficial for localizing the boundaries, as shown in Figure 2.3. Kim et al. performed Hough Transform on the LiDAR data and extracted curbs by finding the longest straight line [20]. Mi et al. generated supervoxels to detect candidate curbs. Afterwards, continuous vectorized road boundaries were produced by a distance clustering strategy [26]. It is worth mentioning that a lot of refinement steps are needed to account for the completeness and accuracy of extracted road boundaries, due to varying road conditions. The refinement is always based on assumptions, such as an equal curvature for the left and right road sides. Also, the boundary of roads without patterns like curbs cannot be effectively extracted in this way.

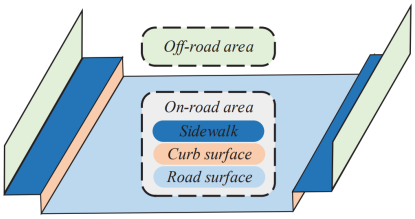


Figure 2.3: Illustration of a road area with curbs [46].

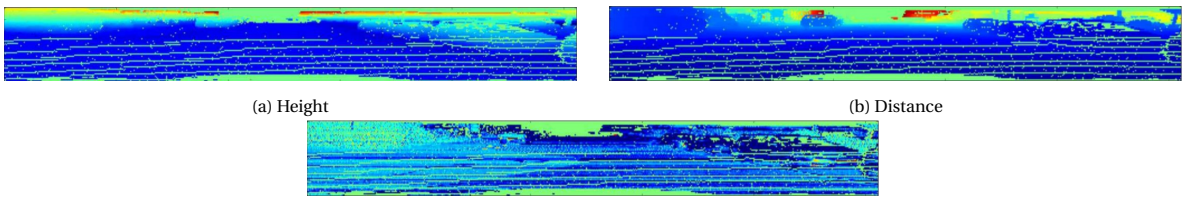
Also, semantic segmentation can be used as an intermediate step for road boundary extraction. Hou et al. created polyline-based inventory of sidewalks from mobile LiDAR data based on the point cloud semantic segmentation results output by a deep neural network, PointNet++ [17]. Obviously, the performance of semantic segmentation is crucial to the quality of boundary extraction in the next step. Since man-made objects (e.g., sidewalk and cycling path) can show similar properties in features like planarity, there remains a lot of challenges in distinguishing different road surface types through the end-to-end detection methods. Moreover, research on classifying road surface into detailed functions using LiDAR point clouds is still lacking.

2.2. Point Cloud Semantic Segmentation by Deep Learning

Recent studies on semantic segmentation of point clouds using deep learning mainly consist of two kinds of methods, i.e., **projection-based** and **point-based** methods [14].

2.2.1. Projection-based Methods

Although image recognition has benefited a lot from deep learning techniques in recent years, it is difficult to apply methods like CNNs directly on point clouds due to their irregular data structure. A common approach to avoid processing unordered 3D points is to project the point cloud data onto 2D planes, i.e., images. Then 2D CNNs can be applied to tasks such as semantic segmentation. The images can be generated from multiple perspectives, with pixel values filled by point attributes (e.g., height and intensity), as indicated by Figure 2.4. Qin et al. classified an airborne point cloud into different terrain scene categories (e.g., plain vegetation, terrace, and rough terrain) by applying a 2D CNN on multi-view images generated from the LiDAR data [30]. Instead of single-view images, Milioto et al. [27] adopted spherical projection to form range images. Semantic labels achieved by a 2D fully convolutional neural network are then converted back to 3D point clouds, with an effective KNN-based post-processing step to reduce discretization errors. Compared to multi-view projection, spherical projection can preserve more information of the LiDAR point cloud [14].



(a) Height (b) Distance

(c) Intensity

Figure 2.4: Front-view images from LiDAR point clouds [7].

Besides 2D representations, converting point clouds into voxels (i.e., 3D grids) can also achieve an aligned data structure before processing with deep learning. Riegler et al. [32] proposed OctNet, a 3D CNN, to realize point cloud semantic segmentation by leveraging a hybrid grid-octree structure (see Figure 2.5). The

point cloud is hierarchically partitioned beforehand, allowing efficient computation on areas with different densities.

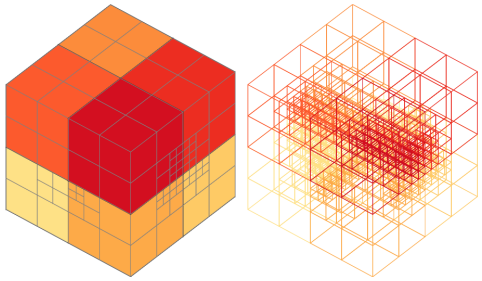


Figure 2.5: Hybrid grid-octree data structure used by OctNet, with the depth of octrees indicated by different colors [32].

Although projection-based methods can address the problem of unorganized data structure of 3D point clouds indirectly, discretization errors and occlusions can be caused during the generation of intermediate representations, i.e., images and voxels. Additional computational resources are also required during pre-processing of the point clouds.

2.2.2. Point-based Methods

As an active research topic in point cloud processing by deep learning, point-based methods enable neural networks to directly consume and model 3D point data. PointNet [28], the pioneering work among these methods, employs a series of shared multi-layer perceptrons (MLP) to learn high-dimensional features for each individual point (see Figure 2.6). Then these per-point features are aggregated globally by applying a symmetric function (e.g., max-pooling), ensuring that point cloud processing is irrelevant to the point order. However, PointNet does not consider local structures inside the point cloud, limiting its performance in complex scenes [29].

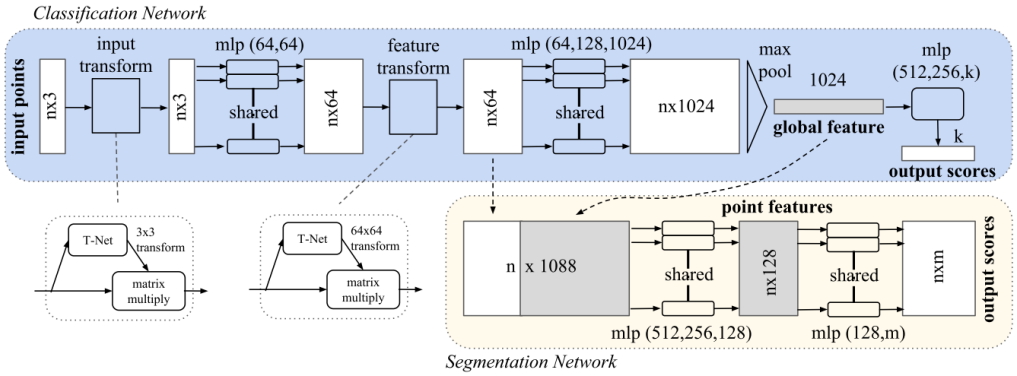


Figure 2.6: Overview of the PointNet architecture [28]. n: number of input points, T-Net: mini-network to transform point coordinates into a canonical space, mlp: multi-layer perceptron, k: number of object classes in the classification task, m: number of semantic labels in the segmentation task.

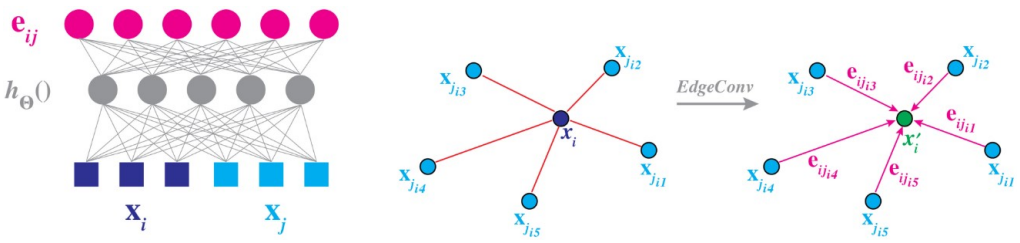


Figure 2.7: Local feature aggregation in DGCNN [42]. **Left**: Edge feature e_{ij} computed from the central point x_i and neighboring point x_j. $h_\Theta()$: a fully connected layer. **Right**: EdgeConv, in which the edge features associated with all neightboring points of x_i are aggregated by a symmetric function.

Starting from PointNet, many networks are proposed, combining MLPs with local feature aggregation. A local feature aggregation module aims to extract prominent features from a point neighborhood, thereby exploiting rich contextual information around each point. Qi et al. proposed PointNet++, which learns local features by applying PointNet on a point neighborhood selected by a ball query with a fixed radius [29]. PointNet++ also achieves hierarchical feature learning by reducing the number of central points in each layer through iterative farthest sampling. DGCNN [42] incorporates local information by first constructing a K-Nearest Neighbor (K-NN) graph from the point cloud. Then, an edge feature between each neighboring point and the central point in a local region is computed through a fully connected layer. As shown in Figure 2.7, edge features associated with the centroid are then aggregated by a symmetric function (e.g., max-pooling or summation) to update features of the central point, which is similar to the convolution operation in 2D CNNs. In each layer, the K-NN graph is generated again based on distances in the complete feature space, resulting in an increasing receptive field while the number of points remains the same after each "convolution" operation. PointCNN [23] also achieves point convolution by learning a so-called χ-transformation from the input points. The points in a local neighborhood are weighted and permuted after the transformation, followed by element-wise product and sum operations of the typical convolution operator. Liu et al. proposed a Relation-Shape Convolutional Neural Network (RS-CNN) to obtain contextual shape-aware learning of 3D point clouds [24]. In this network, the convolutional weights are learned from a predefined geometric relation vector, e.g., a higher-dimensional feature transformed from the 3D Euclidean distance between a neighboring point and the centroid, to model the topology of each local region. Unlike networks that imitate the convolution operation by constructing point neighborhoods using K-NN search or ball query, SPLATNet [36] interpolated the raw 3D points to a higher-dimensional permutohedral sparse lattice and achieved standard spatial convolutions on occupied parts of the lattice. The filtered output is interpolated back to the original point cloud in the end.

During local aggregation, the choice of input features has great impact on the effectiveness. PointNet++ adopts relative coordinates in a local region together with additional point features (e.g., color), while DGCNN uses the concatenation of all original and relative features as input. RandLA-Net [18], similar to RS-CNN, employs a more complex encoding for relative coordinates to capture geometric details in the local neighborhood. The encoded geometric feature vector, together with additional point features, is then used to achieve local feature aggregation. Previous studies evaluating different input features of the local aggregation module suggest that one fixed feature combination is not optimal for all datasets [25].

2.3. Improvement of Delineation in Semantic Segmentation

In semantic segmentation of images, label predictions output by CNNs can be ambiguous near boundaries between different types of objects. This is partly caused by the convolution operation with a large receptive field to extract high-level features of the whole image. Downsampling operations such as max-pooling can also discard low-level details, leading to coarse outputs in per-pixel labeling [45]. This non-sharp delineation problem exists in point cloud semantic segmentation as well, as shown in Figure 2.8. Given a point set, the local information needs to be extracted from a point neighborhood. In some deep learning methods, a symmetric function (e.g., max-pooling) applied to the local neighborhood avoids dealing with the point order, but inevitably treats the neighboring points indistinguishably. In this case, point features from different objects can contribute to the labeling of boundary points with no difference. Deep neural networks help to gain wider contextual knowledge of the point cloud by progressively aggregating local features through a series of layers, which is important to scene understanding. However, accurate semantic segmentation also requires low-level but high-resolution information to ensure the consistency of point label assignments near boundaries [47].

Probabilistic graphical models such as Markov Random Fields (MRF) have been widely used in computer vision to improve the performance of both image and point cloud semantic segmentation. As a specific case of MRF, Conditional Random Field (CRF) formulates the per-pixel (or per-point) labeling task as a statistical inference problem based on the assumption that pixels (or points) showing similar features, e.g., color and intensity, tend to have the same label. Chen et al. added a fully connected CRF at the final layer of a deep CNN to enhance the accuracy of image semantic segmentation [6]. Zheng et al. implemented mean-field inference of CRF with Gaussian pairwise potentials as a recurrent neural network (RNN) and embedded it in existing CNNs [47], achieving end-to-end training of the CNN and CRF-RNN. The weights of the CNN can

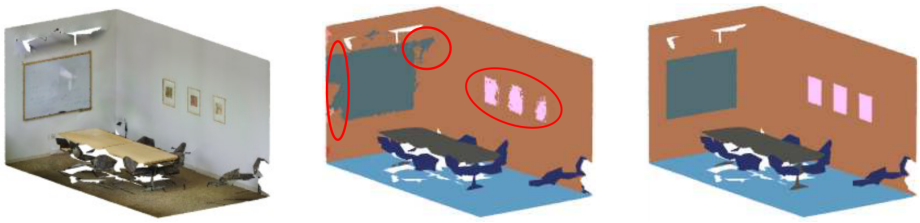


Figure 2.8: Illustration of ambiguous delineation point cloud semantic segmentation [41]. **Left:** input point cloud. **Middle:** prediction. **Right:** ground truth.

then be adapted to the behavior of the CRF module. While high-dimensional features learned from CNNs are significant in determining semantic labels, the appearance and spatial consistency of pixels are important for achieving precise predictions. CRF shows its effectiveness by adding this information back to encourage the label agreement between similar pixels. In 3D cases, SEGCloud applied the CRF-RNN to optimize coarse voxel prediction of a 3D CNN [38]. SqueezSeg refined road-object segmentation results from LiDAR point clouds using a combination of 2D CNN and CRF-RNN by first projecting the point cloud onto a sphere [45].

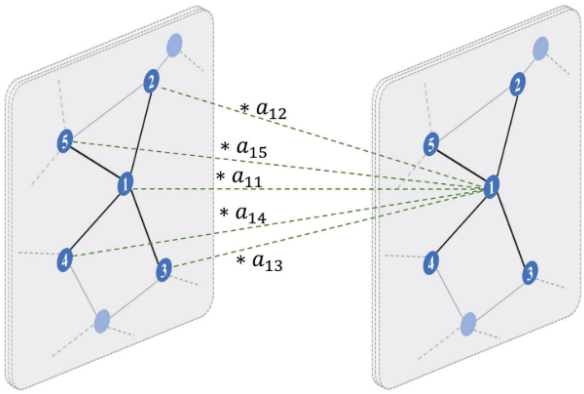
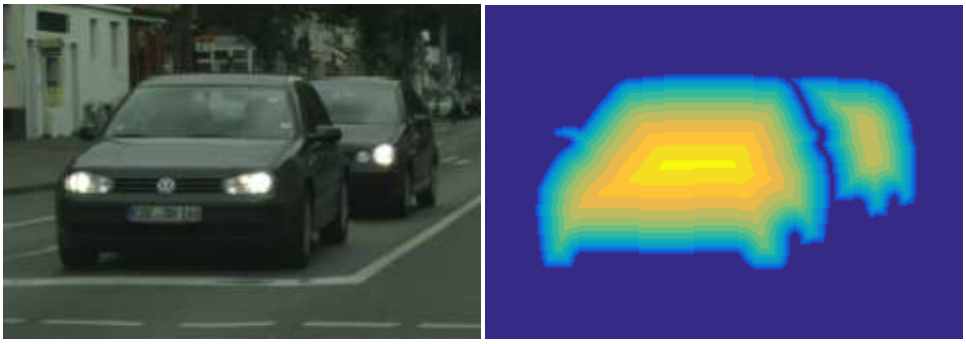


Figure 2.9: Illustration of the Graph Attention Convolution (GAC) operation on a subgraph of a point cloud [41]. The output of GAC is a weighted combination of neighboring points of point 1. Note that point 1 itself is also included in the neighborhood.

Considering that 3D point clouds can be represented as graphs, with points as graph nodes, some point-based networks try to improve the delineation of point cloud semantic segmentation by using a so-called graph attention operation. Theoretically, such attention mechanism allows the network to focus on the most relevant part of the input data, which can be well applied to graph representations (e.g., K-NN graphs) and help identify the most relevant neighboring point features for a central point, creating a *masking* effect for points from another semantic type [39]. Wang et al. [41] designed a Graph Attention Convolution (GAC) operation (see Figure 2.9) to learn a weight matrix acting on the point neighborhood based on the feature difference between the central point and its neighboring points, achieving better results than CRF-based methods.

In image instance segmentation, methods for improving the delineation between objects have also been proposed. Instance segmentation requires pixel-level (or point-level) classification while also aims at segmenting each instance within one class [16]. Bai et al. [1] learned the so-called watershed energy of different objects in an image through a deep neural network to achieve accurate instance segmentation. As shown in Figure 2.10, the level of watershed energy indicates the distance from one foreground (i.e., car) pixel to the nearest boundary. The watershed energy becomes lower when moving from the middle of objects to boundaries. Energy values on and outside the boundary are zero. The final prediction of foreground objects can be produced through cutting the image at a single watershed energy level. Bischke et al. [4] also adopted energy levels based on distances to achieve better semantic segmentation of aerial imagery using CNNs. In order to classify the pixels into *building* and *non-building*, they conducted multi-task learning to output semantic labels and energy level predictions simultaneously. Since the energy levels indicate distances to boundaries, the building boundary information is incorporated in deep neural networks and can be regarded as a con-

straint for semantic labeling.



(a) Input image (b) Watershed energy

Figure 2.10: Illustration of the watershed energy [1].

As mentioned above, strategies for enhancing the delineation in segmentation results are mainly proposed in the image domain. As for point cloud semantic segmentation, the ambiguous boundary issue is also commonly seen, especially in deep learning methods. Some refinement approaches for point clouds depend on first projecting the points as images or voxels to use methods such as CRF, requiring additional pre-processing steps. Attention mechanism adopted in point-based neural networks can alleviate the ambiguous delineation to some extent, but lacks explicit constraints to ensure local consistency in label predictions. Improvement of delineation in point cloud semantic segmentation still needs more research.

3

Methodology

This chapter presents the methodology used in this project, including data pre-processing (Section 3.1), road type classification of MLS point clouds using RandLA-Net as well as strategies to improve the classification performance near boundaries (Section 3.2), 2D boundary polygon extraction through post-processing (Section 3.3), and evaluation (Section 3.4). The main workflow is illustrated in Figure 3.1.

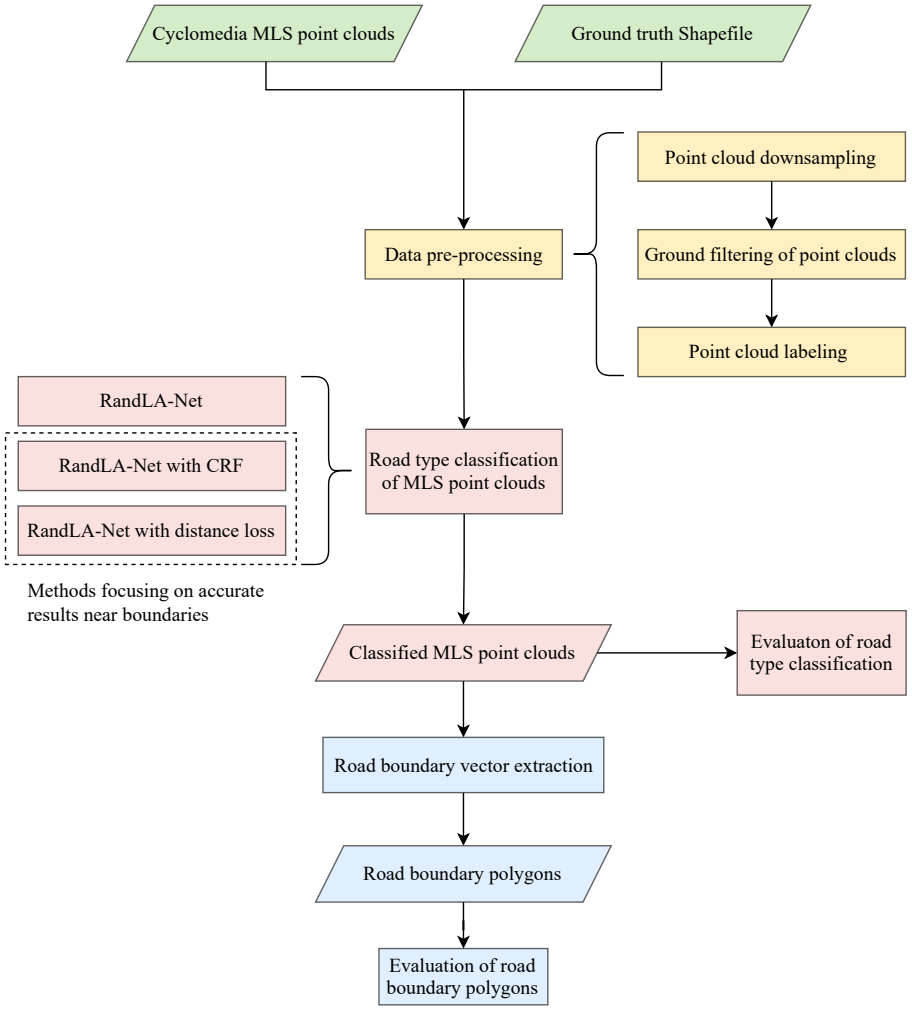
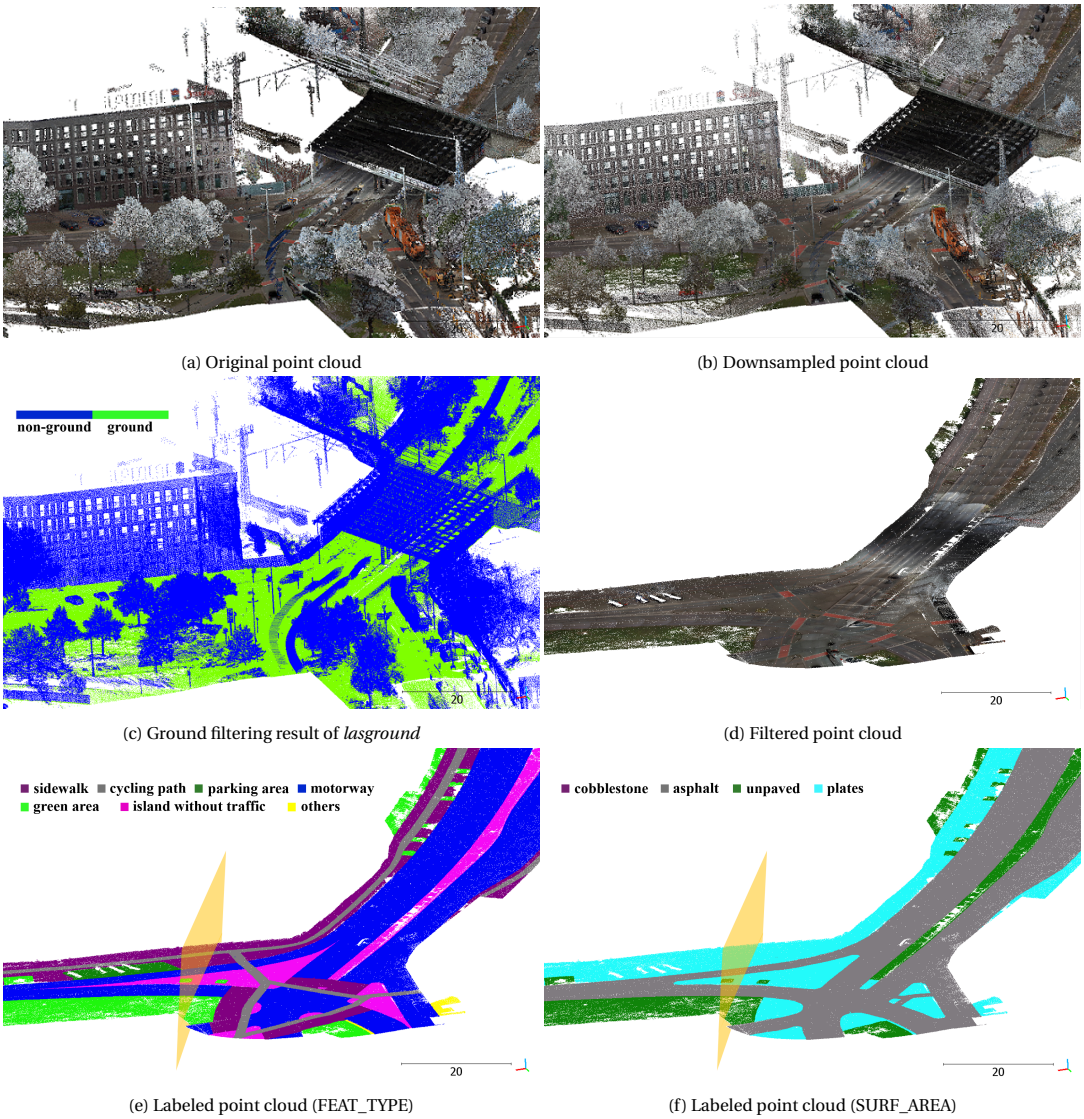


Figure 3.1: Workflow of road boundary mapping in this study, with main components represented in different colors.

3.1. Pre-processing Steps

To handle the sheer volume of acquired point clouds, we first downsample them through grid sampling, with a grid size of 0.1 m. Figure 3.2a and 3.2b present comparison between the colorized point clouds before and after downsampling, respectively. Since objects above the ground (e.g., trees and buildings) are not relevant to road surface type classification, the *lasground* tool[1] is applied to remove points from non-ground objects and noise caused by moving objects on the road. Figure 3.2c shows a ground filtering result output by *lasground*. By filtering out the non-ground points, a ground-level road point cloud is obtained (see Figure 3.2d).



(a) Original point cloud

(b) Downsampled point cloud

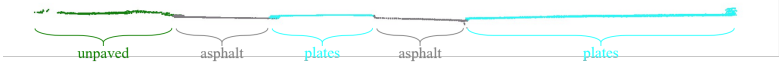
(c) Ground filtering result of *lasground*

(d) Filtered point cloud

(e) Labeled point cloud (FEAT_TYPE)

(f) Labeled point cloud (SURF_AREA)

(g) Labeled cross-section profile (FEAT_TYPE)



(h) Labeled cross-section profile (SURF_AREA)

Figure 3.2: Point cloud pre-processing steps. The original point clouds are firstly downsampled to reduce the data volume. Afterwards, non-ground objects are filtered out by *lasground*. Finally, road point clouds are labeled with FEAT_TYPE and SURF_AREA.

[1] http://rapidlasso.com/LAStools

To label the filtered point clouds, we utilize the ground truth annotations in the Shapefile format, which document road surface types as polygons. Specifically, points contained in each ground truth polygon are searched using the *within* function implemented in the Shapely Python package[2]. The labeling results are indicated by Figure 3.2e and 3.2f. Labels FEAT_TYPE and SURF_AREA refer to the usage type and material type of road surfaces, respectively. We also provide the cross-section profile of the labeled points in Figure 3.2g and 3.2h, which are acquired by cutting the point clouds along the z-axis, as illustrated by the orange plane in Figure 3.2e and 3.2f. Through the cross-section profile, we can observe elevation changes of the road surface. The largest height difference in Figure 3.2g is approximately 45 cm. Moreover, point clouds in each pre-processing step are stored in the LAS format.

3.2. Road Type Classification of MLS Point Clouds by Deep Learning

We implement RandLA-Net for road type classification in this study[3]. RandLA-Net is designed for the semantic segmentation of large-scale point clouds. To achieve efficient processing, random point sampling rather than complex point selection methods (e.g., farthest point sampling) is adopted in RandLA-Net [18]. However, like other deep learning methods, accurate delineation in the segmentation results can be hard to acquire with RandLA-Net due to the loss of low-level details during downsampling. Although the skip connection used in RanLA-Net simply adds low-level features back to the final feature map, it does not ensure accurate delineation for adjacent objects that belong to different classes but have indiscriminate features. To improve the road type classification performance near boundaries, we propose two strategies. The first method is to implement a so-called conditional random field (CRF) as a recurrent neural layer and connect it to the last layer of RandLA-Net. CRF takes the output of RandLA-Net and original point features (e.g., x, y, z, R, G, B) as input to enhance the local consistency of predicted labels. In the second approach, we compute the distance from each point to its nearest boundary. The boundary information can be learned by the network through encoding the distance value as another label and adding a loss function related to the distance label. By such multi-task learning, we apply a certain geometric constraint to RandLA-Net to obtain better road type classification results near boundaries.

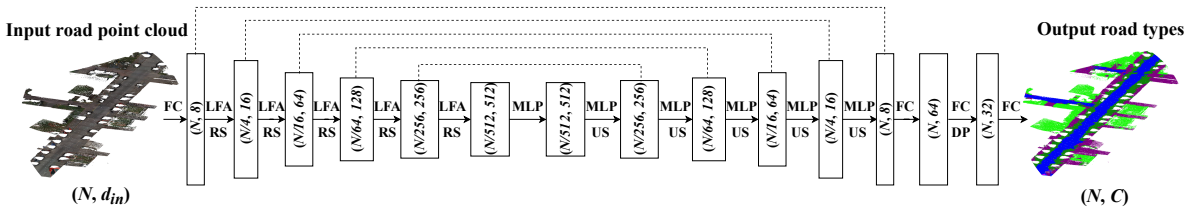


Figure 3.3: Overview of RandLA-Net architecture, with rectangles representing the dimension of points during processing. N: Number of input points, d_{in}: Input feature dimension, FC: Fully Connected layer, LFA: Local Feature Aggregation module, RS: Random Sampling, MLP: Multi-layer Perceptrons, US: Up-sampling, DP: Dropout, C: Number of output classes.

3.2.1. RandLA-Net

RandLA-Net is a point-wise neural network and follows an encoder-decoder hierarchical design with skip connections. Figure 3.3 illustrates the RandLA-Net architecture adopted in this study. Given a point cloud with a large number of points, the points are first progressively downsampled with random sampling in each encoding layer to increase the receptive field. As shown in Figure 3.3, the original point cloud is downsampled for 5 times, with ratios 4, 4, 4, 4, and 2. Afterwards, the encoded points are upsampled again in decoding layers to preserve the original resolution in final predictions.

Since random sampling drops points non-selectively, an effective local feature aggregation (LFA) module is also designed in each encoding layer to summarize neighborhood information without losing important point features. The neighborhood around each point is selected using K-Nearest Neighbor (KNN) in RandLA-Net. The red rectangle in Figure 3.4 illustrates the changing extent of such a point neighborhood during the

[2] https://github.com/Toblerity/Shapely

[3] We select the network from PointCNN, GACNet, and RandLA-Net through testing their performance on our dataset. Details can be found in Appendix A.

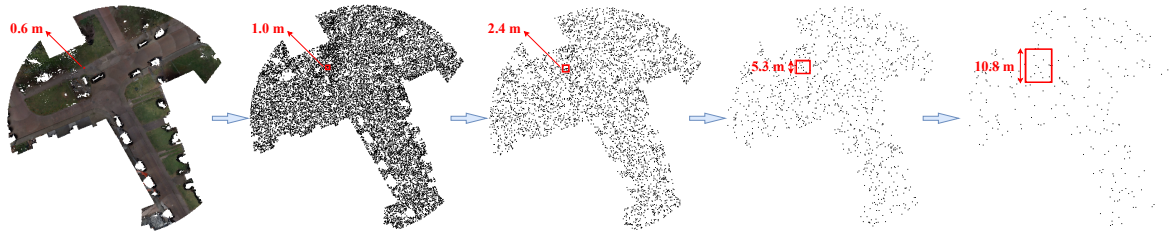


Figure 3.4: Point clouds from the top view that are input to each LFA module of RandLA-Net. Red rectangles indicate the changing extent of K nearest neighbors, with $K = 16$. The original number of points N is 65536 and the original point interval is 0.2 m.

processing of encoding layers. With $K = 16$, the local region fed into the LFA module in each layer is enlarged progressively due to random sampling, resulting in a growing receptive field.

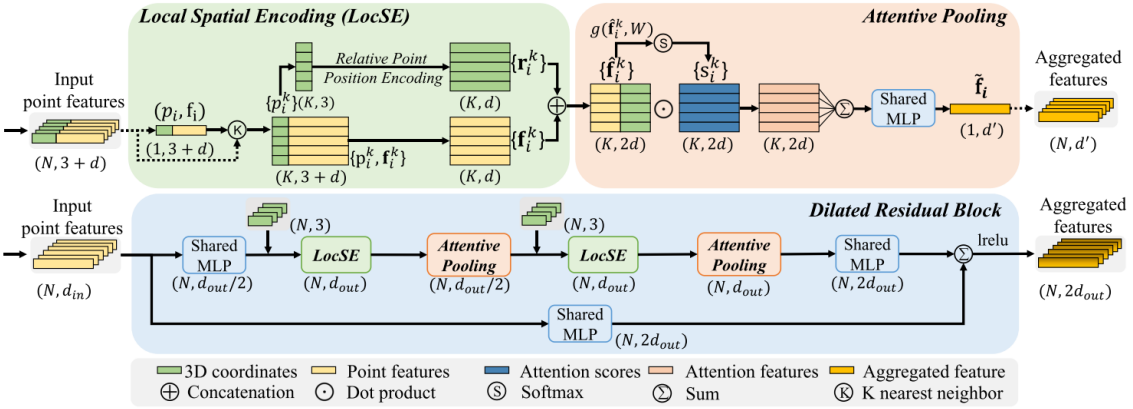


Figure 3.5: Components of an encoding layer in RandLA-Net. **Top**: Local Spatial Encoding (LocSE) block which transforms the input features and Attentive Pooling block which aggregates the local information based on weighing the neighboring points. **Bottom**: Two pairs of LoSE and Attentive Pooling blocks are stacked together to increase the receptive field, which forms the Dilated Residual Block of each encoding layer [18].

Moreover, the LFA module is key to modeling and perceiving the local geometry of point clouds. As shown in Figure 3.5 (top), local feature aggregation in RandLA-Net is mainly composed of two steps, i.e., Local Spatial Encoding (LocSE) and Attentive Pooling. Within LocSE, coordinates of the input points are first transformed to a higher dimensional geometric feature vector r_i^k according to:

$$r_i^k = MLP(p_i \oplus p_i^k \oplus (p_i - p_i^k) \oplus \|p_i - p_i^k\|), \tag{3.1}$$

where
MLP = multi-layer perceptrons
$i \in \{1, 2, ..., N\}$
N = the total number of points
$k \in \{1, 2, ..., K\}$
K = the number of nearest neighbors
p_i = coordinates of the centered point
p_i^k = coordinates of one neighboring point
$\oplus$ = concatenation operation
$\|\cdot\|$ = Euclidean distance

The geometric feature vector r_i^k and additional point features f_i^k (e.g., R, G, B) are then concatenated as $\hat{f}_i^k$, which is the input of Attentive Pooling. Attentive Pooling also borrows the idea of the attention mechanism introduced in Section 2.3, which aggregates the enhanced point feature $\hat{f}_i^k$ in the neighborhood to achieve local contextual information for each point. Networks such as PointNet++ [29] and DGCNN [42] apply a symmetric function (e.g., max-pooling and $\sum$) as the aggregation function, which is simple, but inevitably processes the neighboring points indistinguishably, causing a certain loss of geometric information. The Attentive Pooling in RandLA-Net, instead, learns different weights s_i^k of the neighboring points through an

MLP, as indicated by $g(\hat{f}_i^k, W)$ in Figure 3.5 (top). The neighborhood features are subsequently aggregated by taking a weighted sum. Also, RandLA-Net applies the LFA module twice in each encoding layer to effectively increase the receptive field of the network, as shown in Figure 3.5 (bottom).

3.2.2. CRF as RNN Connected to RandLA-Net

As a probabilistic graphical model, CRF has been widely applied to refine coarse predictions in image semantic segmentation to produce sharp boundaries [47]. In the image domain, a CRF formulates each pixel label as a random variable that is related to information provided by the whole image. Given an image $\mathbf{I}$ and corresponding random variables $\mathbf{c} = \{c_1, c_2, ..., c_N\}$ that refer to the pixel labels, with N the number of pixels in the image, the relation between $\mathbf{I}$ and $\mathbf{c}$ is described by a CRF model:

$$P(\mathbf{c} = c|\mathbf{I}) = \frac{1}{Z(\mathbf{I})\exp\left(-E(c|\mathbf{I})\right)}, \tag{3.2}$$

where $Z(\mathbf{I})$ denotes the partition function, which is used for normalization. Moreover, c is a possible value taken from the predefined labels $L = \{l_1, l_2, ..., l_k\}$, with k the number of semantic classes. $E(c)$ is the energy of a pixel assigned to the label c, which is given by:

$$E(c|\mathbf{I}) = \sum_i u_i(c_i) + \sum_{i,j} p_{i,j}(c_i, c_j), \tag{3.3}$$

in which the unary energy term $u_i(c_i)$ measures the cost of assigning label c_i to pixel i, which can be obtained from a classifier such as CNN. The pairwise energy term $p_{i,j}(c_i, c_j)$ defines the cost of labeling pixels i, j as c_i, c_j at the same time. $p_{i,j}(c_i, c_j)$ can add a penalty when assigning different labels to a pair of pixels with similar features [47]. The pairwise energy term is typically modeled as a weighted sum of Gaussian kernels:

$$p_{i,j}(c_i, c_j) = \mu(c_i, c_j) \sum_{m=1}^{M} w_m k^m(\mathbf{f}_i, \mathbf{f}_j), \tag{3.4}$$

where k^m refers to the m-th Gaussian kernel which is applied to features $\mathbf{f}$ of pixels i and j, e.g., pixel locations and RGB values. w_m denotes the weight coefficient. Moreover, $\mu(c_i, c_j)$ depicts the label compatibility, which equals 1 if $c_i \neq c_j$ and 0 otherwise. Krähenbühl et al. [21] proposed to use two contrast-sensitive Gaussian kernels for multi-class image segmentation, which are defined as:

$$\underbrace{w_1 \exp\left(-\frac{|p_i - p_j|^2}{2\alpha^2} - \frac{|I_i - I_j|^2}{2\beta^2}\right)}_{\text{appearance kernel}} + \underbrace{w_2 \exp\left(-\frac{|p_i - p_j|^2}{2\gamma^2}\right)}_{\text{smoothness kernel}}, \tag{3.5}$$

where p_i and p_j are the pixel positions described by their row and column indices. I_i and I_j denote the color features of pixel i and j. In addition, The first kernel is called *appearance kernel*, which is based on the assumption that nearby pixels with similar appearance (i.e., color) tend to have the same label. α and β in Equation 3.5 control the importance of closeness and color similarity. The second term adjusted by γ is called *smoothness kernel*, which helps to eliminate small spurious regions in the segmentation results. Apparently, the CRF model described above can also be applied to 3D point clouds if we substitute p_i and p_j with coordinates (x, y, z) in a required reference system. Besides RGB values, I_i and I_j can also consist of LiDAR features such as intensity.

Refining the label predictions in the image or point cloud semantic segmentation results through CRF requires to minimize the energy function in Equation 3.3, which is approximately achieved by a mean-field iteration method as proposed in [21]. Instead of computing the exact distribution as shown in Equation 3.2, the mean-field method approximates $P(\mathbf{c})$ using a product of independent marginals, i.e., $Q(\mathbf{c}) = \prod_i Q_i(c_i)$. Then the CRF distribution $P(\mathbf{c})$ is approximately derived by minimizing the KL-divergence $\mathbf{D}(Q||P)$, which is solved in an iterative manner in the mean-field algorithm. After initializing Q, each iteration consists of four steps, i.e., **message passing**, **compatibility transform**, **unary update**, and **normalization**.

As shown in Algorithm 1, during the **initialization** of the mean-field algorithm, $Q_i(c_i)$ is initialized by applying a softmax function over the unary terms $-u_i(c_i = l)$, with $Z_i = \sum_l \exp\left(-u_i(c_i = l)\right)$. The **message passing** operation consists of applying all the Gaussian filters on Q. Coefficients of Gaussian filters are calculated

based on features such as pixel (or point) locations and RGB values. **Compatibility transform** further filters the output in the previous step with weights and apply the label compatibility function μ. Moreover, in the **unary update** step, the unary components are added back to Q. Finally, Q is normalized at the end of each iteration.

Algorithm 1: Mean-field algorithm in fully connected CRFs [21].

$Q_i(c_i = l) \leftarrow \frac{1}{Z_i}\exp\{-u_i(c_i = l)\}$ for all i ▷ **Initialization**

while *not converged* **do**

$\tilde{Q}_i^m(l) \leftarrow \sum_{j \neq i} k^m(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ **Message Passing** from all pixel j to i

$\hat{Q}_i(l) \leftarrow \sum_{l' \in L} \mu(l, l') \sum_m w_m \tilde{Q}_i^m(l)$ ▷ **Compatibility Transform**

$\check{Q}_i(l) \leftarrow -u_i(c_i = l) - \hat{Q}_i(l)$ ▷ **Unary Update**

$Q_i \leftarrow \frac{1}{Z_i}\exp\left(\check{Q}_i(l)\right)$ ▷ **Normalization**

end

Zheng et al.[47] formulated one iteration of the mean-field algorithm as a stack of CNN layers, in which parameters in the CRF model such as weights of Gaussians are learned during training. Naturally, multiple iterations can be regarded as a recurrent neural network (RNN), since each iteration takes the output Q values of the previous iteration and the original unary terms (i.e., label predictions of the main classifier) as input. Figure 3.6 illustrates the network structure of such a CRF-RNN, which is applied after RandLA-Net in this study to improve the local consistency of labeling, especially near boundaries.

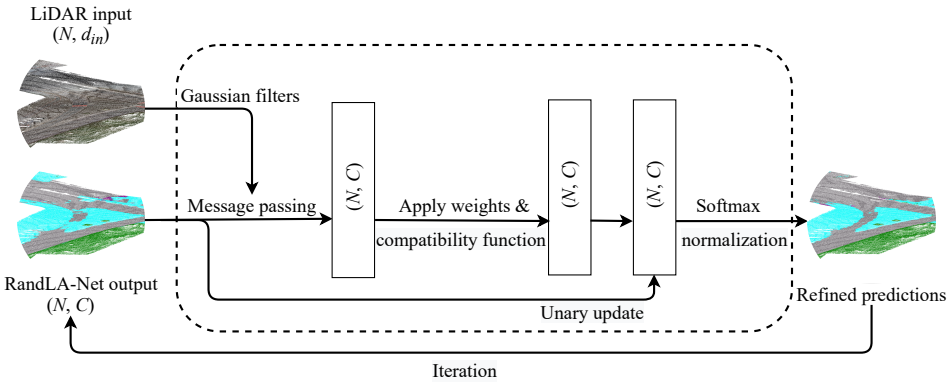


Figure 3.6: Conditional Random Field (CRF) as a Recurrent Neural Network (RNN), with rectangles denoting dimensions of the point cloud data. N: Number of input points, d_{in}: Input feature dimension, C: Number of output classes.

3.2.3. Multi-task Learning of RandLA-Net with Distance Loss

Besides adopting CRF-RNN, we also propose to improve the point cloud semantic segmentation results of RandLA-Net using multi-task learning. In the original implementation of RandLA-Net, the learning depends only on semantic information provided by ground truth labels. After multiple times of local feature aggregation and downsampling, ambiguous predictions near boundaries are commonly seen. To "sharpen" the boundaries in segmentation results, we represent the boundary information related to each point as another label and incorporate it into the loss function during training.

Inspired by distance transform of images [5], which achieves the distance from each pixel to its nearest object boundary, we also calculate distance values for MLS point clouds using the ground truth polygon annotations. Since the point cloud data after pre-processing only contains ground-level road objects, as explained in Section 3.1, 2D distances differ not much from 3D distances within an object. Therefore, we search for all road points within each polygon and obtain distance values through the function *distance* in the Shapely package, which compares (x, y) coordinates of these points with the exterior ring of the corresponding polygon. As shown in Figure 3.7b, the obtained value of each point measures the distance to the nearest boundary, implying the location of boundaries.

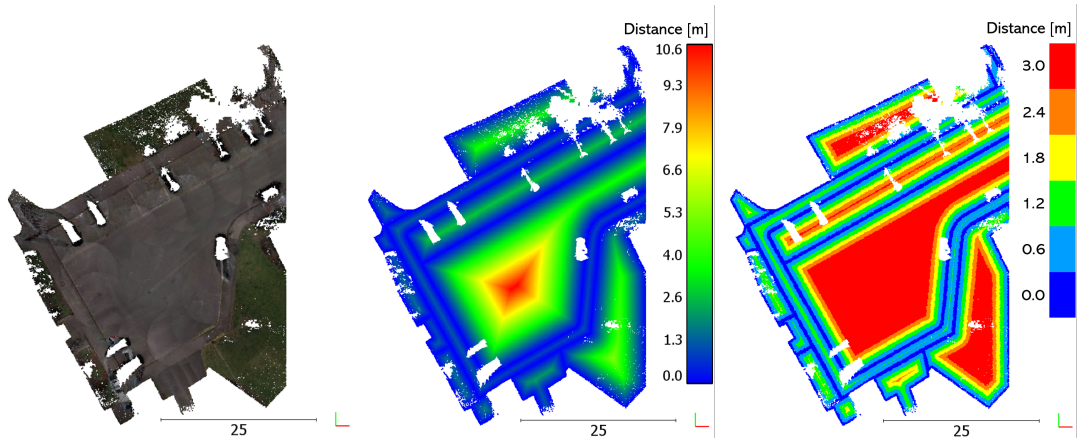
Due to the variation of urban scenes described by the point cloud dataset in this study, the width of a road object (e.g., *motorway*) can vary a lot among different cities and even different locations within one city. To make the distance representation of each point more robust and ease the labeling task, we further convert the original distance values into distance labels following the truncated discrete distance encoding for images in [15]. First, a truncation threshold R, which is the largest distance we care about, is determined to help the network focus on points near boundaries. For each point p, we calculate a truncated distance $D(p)$ according to

$$D(p) = \min\big(d(p), R\big), \tag{3.6}$$

where $d(p)$ is the original distance value of point p. Then, we generate $(M+1)$ discrete labels by quantizing the threshold R into M bins as

$$L_{dist} = \frac{\{0, 1, ..., M\}}{M} R. \tag{3.7}$$

Finally, the distance label of point p is derived by finding its closest value in L_{dist} compared to the truncated distance $D(p)$, as illustrated in Figure 3.7c.



(a) Colorized view (b) Original distance values (c) Distance labels

Figure 3.7: Illustration of distance values and discrete distance labels for an point cloud from the top view. When generating distance labels, truncation threshold $R = 3.0$ m and number of bins $M = 5$ are used.

In this way, it is possible to train the network on two point cloud semantic segmentation tasks, i.e., road type classification and distance prediction, simultaneously (see Figure 3.8).

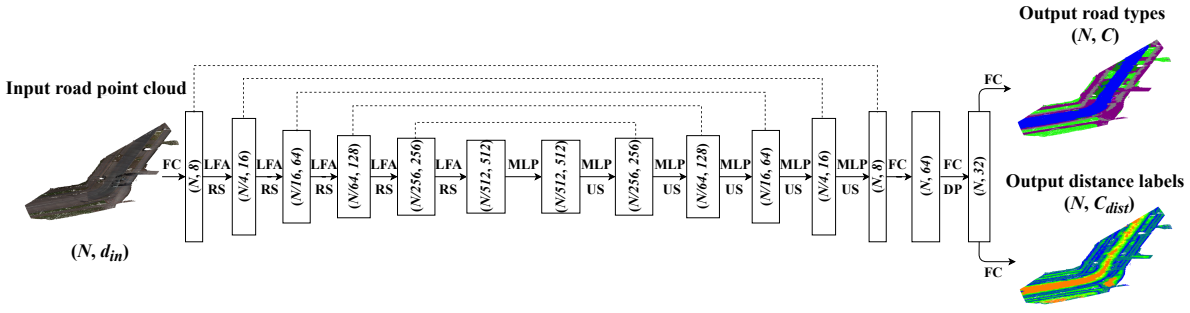


Figure 3.8: Multi-task learning based on RandLA-Net. N: Number of input points, d_{in}: Input feature dimension, FC: Fully Connected layer, LFA: Local Feature Aggregation module, RS: Random Sampling, MLP: Multi-layer Perceptrons, US: Up-sampling, DP: Dropout, C: Number of output classes, C_{dist}: Number of distance labels.

Additionally, we achieve a single loss function during training by combining loss functions of two tasks simply as

$$Loss = Loss_{road\ type} + Loss_{distance}. \tag{3.8}$$

Both $Loss_{road\ type}$ and $Loss_{distance}$ are cross entropy loss. Given prediction scores of each class output by a classifier, the cross entropy loss $\mathscr{L}$ is computed as

$$\mathscr{L} = \frac{\sum_{i=1}^{n} \mathscr{L}_i}{n}, \tag{3.9}$$

where n denotes the total number of points in a batch and $\mathscr{L}_i$ is the loss related to a single point, which is determined by

$$\mathscr{L}_i = -\log\left(\frac{\exp(x_p)}{\sum_j^C \exp(x_j)}\right) = -x_p + \log\left(\sum_j^C \exp(x_j)\right), \tag{3.10}$$

in which x_j represents the prediction score, with $j \in 1,2,...,C$, where C is the total number of classes. Moreover, x_p represents the prediction score for the ground truth class.

3.3. Road Boundary Vector Extraction

After achieving labels of each point, the ground-level point clouds can be segmented into different road types. Furthermore, boundary vectors of road objects can be acquired through post-processing. Given the classified point clouds, we first aggregate points in each class within a certain distance and generate 2D polygons around each point cluster. Boundary polygons of all road types are then combined and refined to remove unnecessary overlaps between different objects.

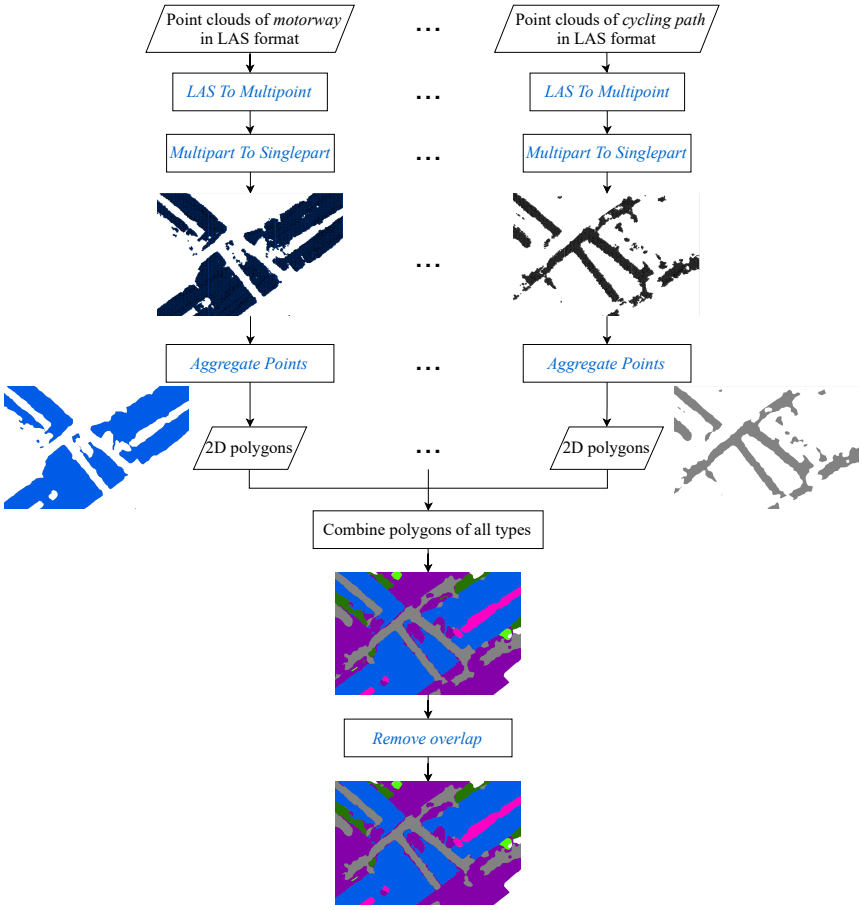
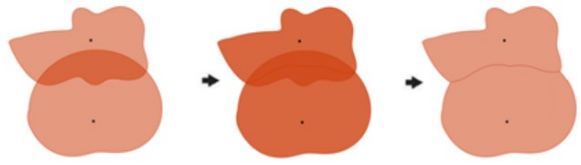


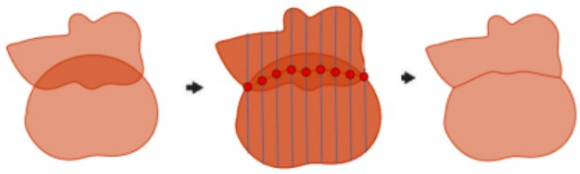
Figure 3.9: Workflow of boundary vector extraction, with texts in blue referring to the tools in ArcGIS Pro.

As shown in Figure 3.9, road boundary extraction is mainly conducted in ArcGIS Pro[4]. The tool *LAS To Multipoint* helps to efficiently load point clouds into ArcGIS Pro as *Multipoint* features. To partition these features into single points, *Multipart To Singlepart* is then applied. Afterwards, given an aggregation distance, the tool *Aggregate Points* can directly create *Polygon* features around clusters of three or more proximate points. The polygons represent the shape described by each point cluster on the road surface (see 2D polygon results in Figure 3.9). However, polygon overlap might be introduced by this approach, resulting in intersecting road boundaries. Having boundary vectors of each class, the next step is to give each polygon a road type attribute and combine polygons in all classes. After the combination, overlaps between different road types can also happen. Therefore, we apply the tool *Remove Overlap* in the end to achieve reasonable topological relations among polygons. In ArcGIS Pro, two overlapping polygons can be refined with three methods (see Figure 3.10):

- **Center Line**, which creates a border that evenly distributes the overlap area between polygons,
- **Grid**, which eliminates the overlap by creating a grid of parallel lines to achieve a natural division,
- **Thiessen**, which divides the overlap area using a straight line.



(a) Center Line



(b) Grid

(c) Thiessen

Figure 3.10: Three options in the *Remove Overlap* tool of ArcGIS Pro[5].

3.4. Evaluation Metrics

In this research, we evaluate both road type classification results of point cloud semantic segmentation and road boundary polygons achieved by post-processing.

3.4.1. Evaluation of Road Type Classification of MLS point clouds

To evaluate road type classification results of MLS point clouds, we determine the following evaluation metrics, which are commonly used in semantic segmentation:

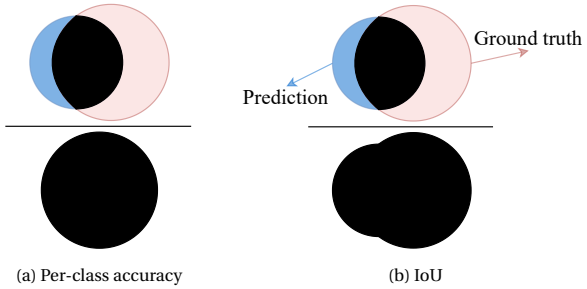
- **Overall accuracy (OA)**, which measures the proportion of correctly classified points among all input points. Since OA ignores the difference between classes, it is not informative enough when there exist class imbalance issues in the dataset.

[4] https://www.esri.com/en-us/arcgis/products/arcgis-pro/overview
[5] Image source: https://pro.arcgis.com/en/pro-app/latest/tool-reference/business-analyst/remove-overlap.htm

- **Mean per-class accuracy (mA)**, which is the average value of the accuracy in each class. Similar to OA, the per-class accuracy measures how many points in a certain class are correctly predicted compared to all points actually belonging to that class.
- **Mean Intersection over Union (mIoU)**, which is the mean value of Intersection over Union (IoU) in each class. Given the predicted mask and ground truth mask of one class, IoU is defined as:

$$\text{IoU} = \frac{\text{Overlap of the predicted and ground truth}}{\text{Union of the predicted and ground truth}}. \tag{3.11}$$

  For point clouds, the overlap and union are calculated based on the number of points. Compared to mA, mIoU emphasizes the similarity between the predicted and ground truth, as illustrated in Figure 3.11.



(a) Per-class accuracy (b) IoU

Figure 3.11: Illustration of per-class accuracy and Intersection of Union (IoU), with black areas representing the values for calculation. Blue circle is the predicted mask and orange circle denotes the ground truth mask.

3.4.2. Evaluation of Road Boundary Vector Extraction

The boundary extraction results of our method are compared to the ground truth annotations based on rasterization. After converting both output and ground truth polygons into raster format, we sample a certain number of points from the images for assessment, since the coverage of output polygons are not completely the same as the ground truth. Then, the predicted and ground truth label of each point are summarized as a confusion matrix, which presents the number of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) in the results. Specifically, if we assume "sidewalk" as the positive class and "not sidewalk" as the negative class, a TP indicates where the model correctly predicts the positive class and an FP refers to where the model incorrectly predicts the positive class. Moreover, a TN is an outcome where the model correctly identify a sample to be "not sidewalk". An FN indicates where the model identifies samples actually belonging to the positive class to be "not sidewalk".

Furthermore, two scores are computed based on the confusion matrix to represent the performance on each road type:

$$\text{precision} = \frac{TP}{TP+FP} \tag{3.12}$$

$$\text{recall} = \frac{TP}{TP+FN}. \tag{3.13}$$

Specifically, recall expresses the ability to detect all relevant instances in the dataset, while precision measures the correct detected instances among all predictions.

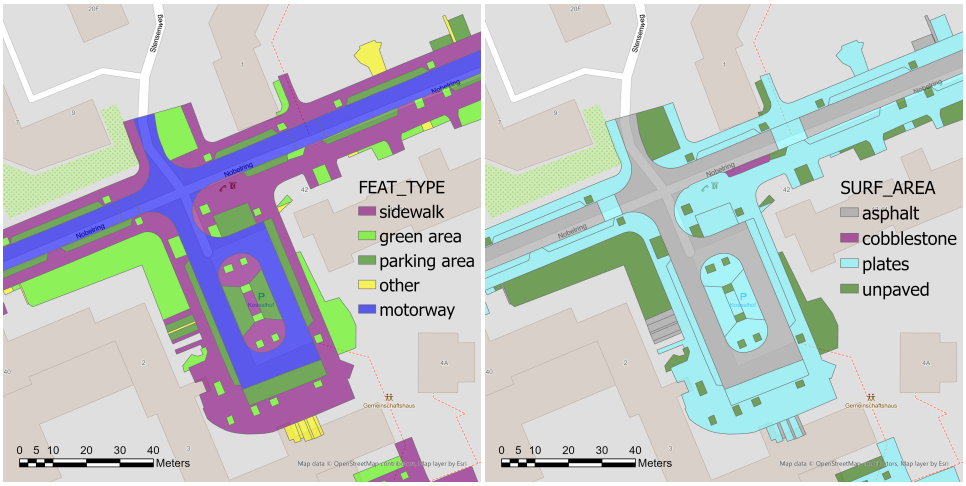
4

Dataset & Study Areas

Section 4.1 describes the Area Mapping product of Cyclomedia, which is used as the ground truth annotation in this project. Section 4.2 presents the properties of MLS point clouds collected by Cyclomedia's mobile mapping system. Afterwards, two study areas with different coverage areas are introduced in Section 4.3 and 4.4, respectively.

4.1. Ground Truth Shapefile Annotations

The Area Mapping product of Cyclomedia, which is adopted as the ground truth base layer, is an inventory of ground-level road types in the public space of urban areas (see Figure 4.1). This product is generated based on orthophotos of Cyclomedia and delivered in the format of ESRI Shapefile, describing road segments using 2D polygons without any overlaps or gaps.



(a) label: FEAT_TYPE (usage)

(b) label: SURF_AREA (material)

Figure 4.1: Examples of ground truth annotations, superimposed on a base map provided by ©OpenStreetMap.

Attribute	Description	Data format
ObjectID	Unique ID of each polygon	integer
Shape_Leng	Length of the road object in meters	double
Shape_Area	Total area of the road object in square meters	double
FEAT_TYPE	Road usage type	text
SURF_AREA	Road surface material type	text

Table 4.1: Data structure of the Area Mapping product of Cyclomedia.

Table 4.1 shows the data structure of the Area Mapping product. Road types are specified by two attributes, i.e., **FEAT_TYPE** and **SURF_AREA**. In this thesis project, both FEAT_TYPE and SURF_AREA are used as labels and required in the final boundary vector product. FEAT_TYPE indicates the usage type of each ground-level road object, which is chosen from 9 classes including

- *sidewalk*: A path designed for pedestrians
- *cycling path*: A bikeway dedicated to cycling and sometimes shared with pedestrians
- *rail track*: A structure enabling the movement of trains or trams
- *parking area*: A space designed for parking in the parking lot or on the city street
- *motorway*: A paved way allowing the travel of motor vehicles
- *green Area*: An unpaved area
- *island without traffic*: A solid or painted road object that channels traffic
- *pedestrian area*: Auto-free zones reserved for pedestrian-only use and usually indicated by traffic signs
- *other*: An area not belonging to the above classes.

The above order of FEAT_TYPE classes reveals the priority of labeling. For instance, if a *cycling path* is crossing a *motorway*, the corresponding road segment should be annotated as *cycling path* instead of *motorway*.

(a) *asphalt* (b) *unpaved* (c) *railway*

(d) *cobblestone*, indicating stones with cobbled size (e) *plates*, indicating stones with rectangular or square size

Figure 4.2: Illustration of road objects with different SURF_AREA types.

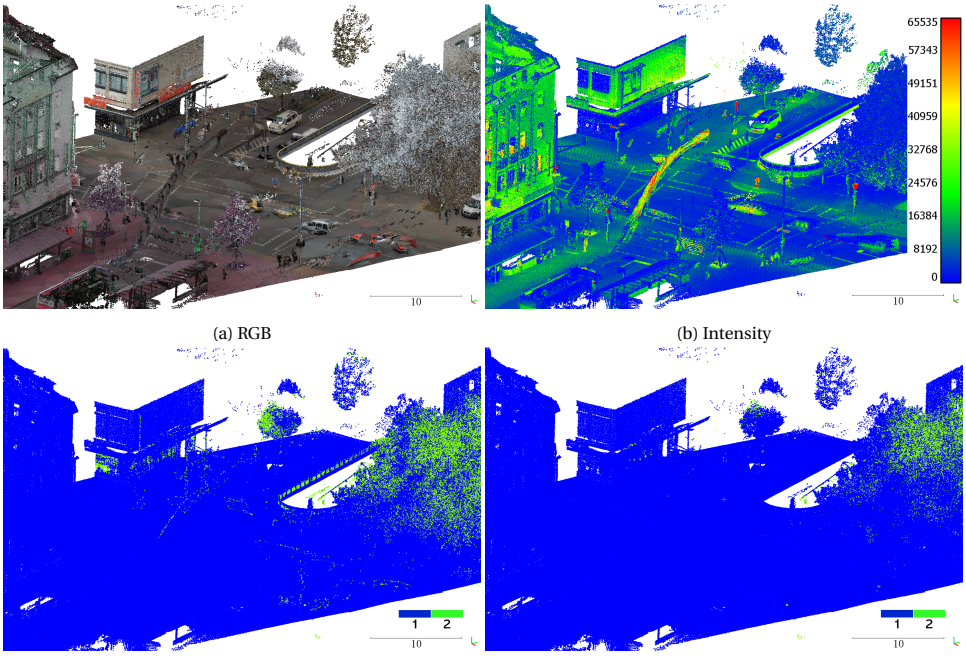
On the other hand, SURF_AREA refers to the material type of road surfaces, including 5 classes, i.e., *cobblestone*, *asphalt*, *plates*, *unpaved*, and *railway* (see Figure 4.2). Note that *railway* in SURF_AREA only refers to structures designed for trains, consisting of rails, fasteners, railroad ties, and ballast. Tramways crossing the city center, as shown in Figure 4.3, are classified as *asphalt* instead.

Figure 4.3: Tramway outside the Rotterdam Centraal railway station[1], which should be classified as *rail track* in FEAT_TYPE but *asphalt* in SURF_AREA.

[1]Image source: https://commons.wikimedia.org/w/index.php?curid=2138027

4.2. MLS Point Clouds from Cyclomedia

The MLS point cloud data provided by Cyclomedia is acquired by a Velodyne HDL-32E LiDAR sensor[2] with an average point spacing of 1 cm. LiDAR features of the point cloud contain intensity, return number and number of returns. Intensity indicates the strength of returned laser beams, which can be used to identify objects with distinct surface properties. As illustrated in Figure 4.4b, points of road markings and some vegetation areas have higher intensity values compared to plain road surfaces. The number of returns refers to the total number of returned signals for a given pulse, while return number is the index of the returned signal related to a LiDAR point. For objects like the tree canopy, multiple laser beams are returned at the receiver, resulting in relatively large values of both attributes (see Figure 4.4c and 4.4d). However, the number of returns and return number of different ground-level objects are not distinguishable. With the Global Positioning System (GPS) antenna and Inertial Measurement Unit (IMU) installed on the recording vehicle, the point cloud from Cyclomedia also holds precise coordinates (i.e., x, y, z) in a corresponding reference system. Moreover, the mobile recording system includes 5 high-resolution cameras that capture panoramic images. As shown in Figure 4.4a, the MLS point cloud is further rendered with Red, Green, and Blue (RGB) colors.



(a) RGB (b) Intensity

(c) Number of returns (d) Return number

Figure 4.4: Attributes of MLS point clouds from Cyclomedia.

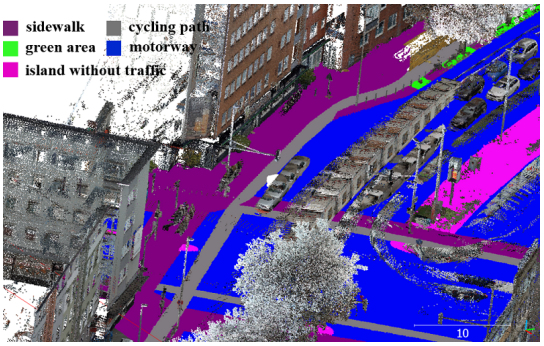
4.3. Study Area 1: Hannover, Germany

As shown in Figure 4.7d, the first study area in this project consists of one recording trajectory from Hannover, Germany. The corresponding point cloud data was collected in April 2020, with points recorded in the ETRS89 coordinate reference system. Data from Hannover has a relatively small quantity and contains all types in FEAT_TYPE and SURF_AREA, making it suitable for extensive experiments on road type classification.

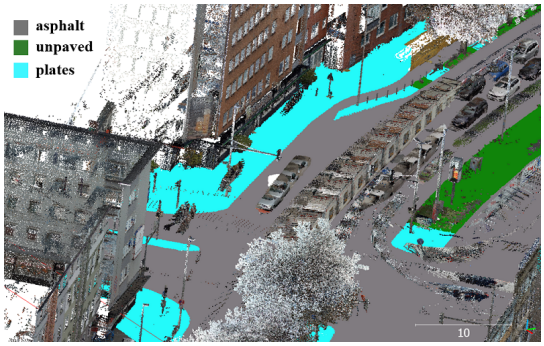
Figure 4.5 shows the point clouds from a crossroad in Hannover. Two kinds of *sidewalk* can be seen from this example, i.e., roadside surface higher than the *motorway* and pedestrian crossings which can be part of the *motorway*. *Cycling path* painted in different colors are also shown. Moreover, the circled area in Figure 4.5a highlights the influence of moving objects on the road during the recording of LiDAR scanners, causing a lot of noise in the MLS point cloud data. Several examples of *pedestrian area* (labeled in orange) and *other* (labeled in yellow) in Hannover are illustrated in Figure 4.6. Obviously, the appearance of *pedestrian area* and *other* can be very similar to *sidewalk*.

[2] https://velodynelidar.com/products/hdl-32e/

(a) Colorized view



(b) FEAT_TYPE



(c) SURF_AREA

Figure 4.5: An example of MLS point clouds from Hannover. The circled area shows the effect of moving objects during recording.

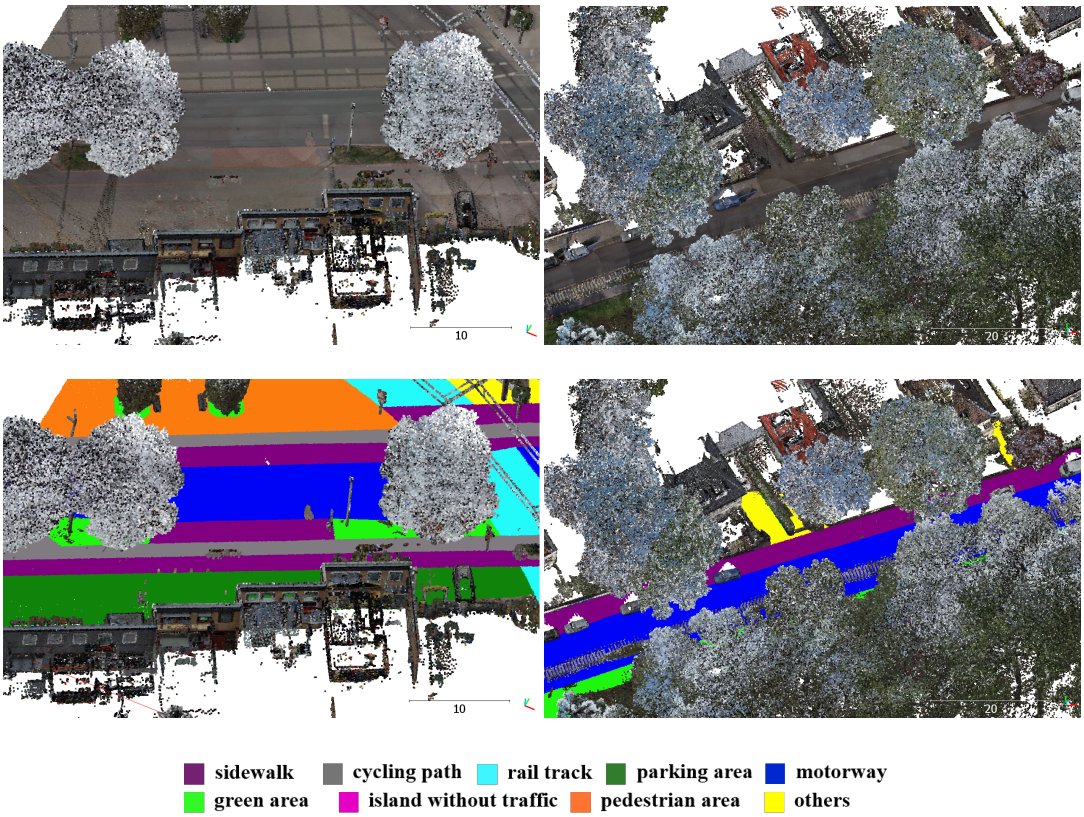


Figure 4.6: *Pedestrian area* and *other* in Hannover. The colors of *pedestrian area* and *other* are similar to *sidewalk* in both examples.

4.4. Study Area 2: 5 German Cities

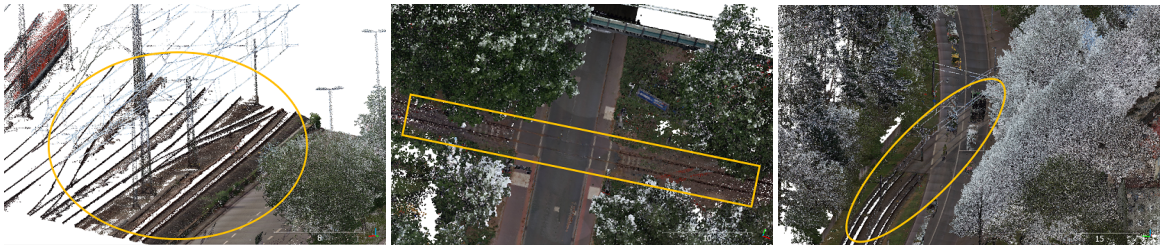
(a) Hamburg (b) Bremerhaven

(c) Delmenhorst (d) Hannover (e) Oldenburg

Figure 4.7: Trajectories (in purple) of the recording vehicle in Hamburg, Bremerhaven, Delmenhorst, Hannover, and Oldenburg, superimposed on a base map provided by ©OpenStreetMap.

Together with Hannover, another four cities from Germany, i.e., Hamburg, Bremerhaven, Delmenhorst, and Oldenburg, form the second study area. As shown in Figure 4.7, the data coverage of Hamburg and Delmenhorst are much larger than Hannover. In addition, although point clouds from 5 cities contain road objects in similar FEAT_TYPE and SURF_AREA types, they show different characteristics in appearance for several specific classes. Compared to Study Area 1, Study Area 2 contains more various road scenes and can be used to test the generalization ability of road type classification models.

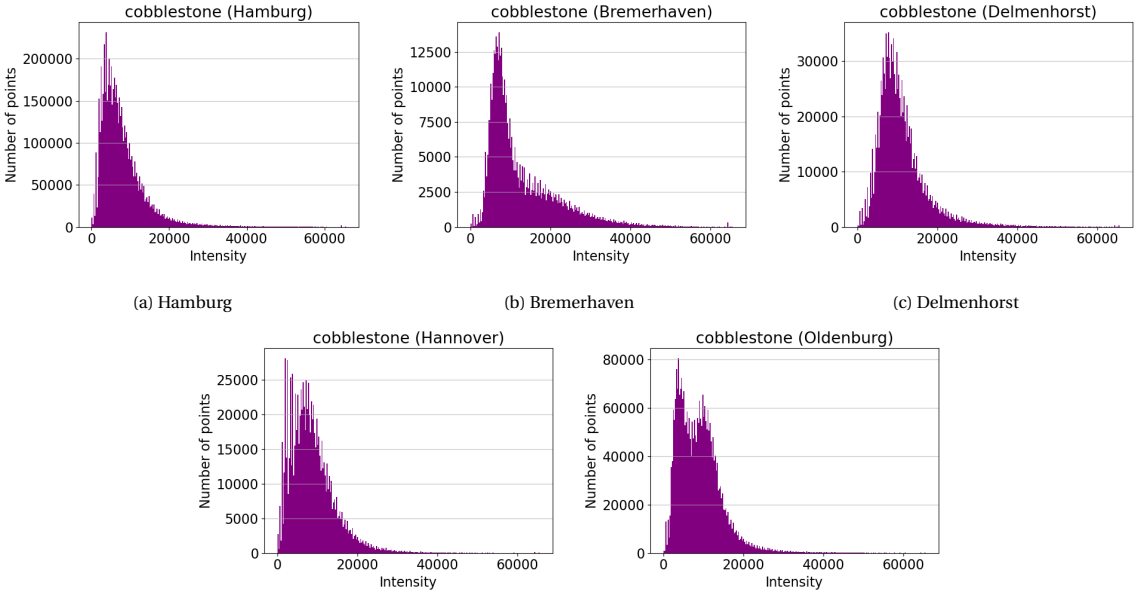
One of the major differences among the 5 cities lies in *rail track* (FEAT_TYPE) or *railway* (SURF_AREA). No *rail track* is recorded in Bremerhaven and Oldenburg. In Hamburg, as shown in the circled area in Figure 4.8a, several parallel rail tracks exist. As a result, rail tracks far away from the recording car are sparsely represented in the point clouds. By contrast, rail tracks in Delmenhorst and Hannover cross the motorway and sidewalks, showing a relatively complete appearance in the point clouds compared to the real objects.



(a) Hamburg (b) Delmenhorst (c) Hannover

Figure 4.8: Rail tracks in Hamburg, Delmenhorst, and Hannover, which show different characteristics.

In the case of SURF_AREA, histograms in Figure 4.9 show that *cobblestone* among 5 cities has different intensity values. In Hamburg, Hannover, and Oldenburg, cobblestones having low reflection (i.e., intensity around 5000) are dominant. From the intensity histogram of Oldenburg in Figure 4.9e we also notice the second peak referring to higher intensity (around 10000), indicating that two types of *cobblestone* can be found in Oldenburg. By contrast, cobblestones in Bremerhaven and Delmenhorst have medium reflection values around 7500, showing different characteristics from the other three cities. In addition, *cobblestone* points having medium intensity can also be observed in Hannover (see Figure 4.9d).



(a) Hamburg (b) Bremerhaven (c) Delmenhorst

(d) Hannover (e) Oldenburg

Figure 4.9: Histograms of *cobblestone* intensity in Hamburg, Bremerhaven, Delmenhorst, Hannover, and Oldenburg.

	sidewalk	cycling	rail track	parking	motorway	green area	island	pedestrian	other	Total
Hamburg	31.19	8.02	1.17	8.40	74.85	14.90	9.19	1.28	11.61	160.61
Bremerhaven	5.92	0.93	0	2.84	11.51	12.28	0.02	0	7.54	41.03
Delmenhorst	20.36	6.55	0.03	6.98	42.94	28.81	0.39	0.01	31.93	138.02
Hannover	15.01	3.22	0.54	3.92	32.68	14.12	2.88	0.26	2.00	74.63
Oldenburg	10.65	2.57	0	4.02	19.70	8.79	0.15	0	11.50	57.39
Total	83.12	21.29	1.74	26.17	181.68	78.91	12.63	1.55	64.58	471.68

Table 4.2: Number of points ($\cdot 10^6$) in each road type (FEAT_TYPE) after pre-processing, with "cycling", "parking", "island", and "pedestrian" indicating *cycling path*, *parking area*, *island without traffic*, and *pedestrian area*, respectively. Points of *pedestrian area* and *other* are merged into the *sidewalk* class for later processing.

Table 4.2 and 4.3 summarize the number of points in each class after pre-processing. From the total amount of points in each FEAT_TYPE class, a class imbalance issue can be observed. *Motorway* contains a dominant number of points. *sidewalk*, *green area*, and *other* are also frequently seen in the data. By contrast, *rail track* has the least amount of points.With regard to SURF_AREA, most road objects are made of *asphalt* and *plates*, which are the common material for *motorway* and *sidewalk*.

Also, as discussed in Section 4.3, points belonging to *pedestrian area* and *others* have a similar appearance as *sidewalk*. Considering that *pedestrian area* and *others* are detected with the help of other information like road signs in practice, both classes are further merged into *sidewalk* to reduce confusion in road type classification.[3] Thus, **7 road types** in FEAT_TYPE, i.e., ***sidewalk***, ***cycling path***, ***rail track***, ***parking area***, ***motorway***,

[3]Compared to using the original implementation of RandLA-Net with separate classes, the mean IoU on Study Area 1 is improved by 11.0% after merging the labels.

	cobblestone	asphalt	plates	unpaved	railway	Total
Hamburg	6.44	88.54	41.37	23.10	1.17	160.61
Bremerhaven	0.45	12.23	14.79	13.56	0	41.03
Delmenhorst	1.20	27.92	76.88	3.10	0.03	138.02
Hannover	0.98	37.24	20.48	15.93	0.003	74.63
Oldenburg	2.94	16.08	28.26	10.10	0	57.39
Total	12.01	182.00	181.78	94.68	1.20	471.68

Table 4.3: Number of points ($\cdot 10^6$) in each road type (SURF_AREA) after pre-processing.

green area, and ***island without traffic*** are finally used in experiments.

5

Results of Road Type Classification of MLS Point Clouds

In this chapter, the results of road type classification of point clouds in Study Area 1 and 2 are presented. Specifically, Section 5.1 shows the classification results using the original implementation of RandLA-Net. Performance with two input feature combinations, i.e., (x, y, z, R, G, B) and (x, y, z, R, G, B, intensity), are compared. Section 5.2 discusses the impact of embedding CRF as RNN in RandLA-Net, especially on the delineation performance in road type classification. Furthermore, the results of using multi-task learning based on RandLA-Net are shown in Section 5.3.

For Study Area 1 (i.e., the Hannover city), the MLS point cloud data after pre-processing is vertically split into 39 tiles, with 29 tiles used for training and 10 for testing. In the case of Study Area 2, we train on point clouds from Hamburg, Bremerhaven, and Delmenhorst. The other two cities (i.e., Hannover and Oldenburg) are adopted for testing to show the generalization ability of our method. Experiments on the label FEAT_TYPE (usage) and SURF_AREA (material) are conducted separately. Moreover, the network is built based on the implementation in Open3D-ML [48] using the deep learning framework PyTorch[1], with the training performed on an NVIDIA Tesla P100 GPU provided by Cyclomedia. During training, a data batch contains 65535 points, which are cropped from the input point clouds, with the central point randomly selected. To account for both accuracy and efficiency, we adopt a point interval of 0.2 m for FEAT_TYPE and 0.1 m for SURF_AREA. Comparisons of using different point densities can be found in Section 6.2. In addition, we use $k = 16$ nearest neighbors around each point to aggregate the local information.

5.1. Results with the Original RandLA-Net Structure

The original implementation of RandLA-Net is evaluated for road type classification on both Study Area 1 and 2. The MLS point cloud data has 9 attributes, i.e., coordinates (x, y, z), color (R, G, B), and LiDAR features (intensity, return number, number of returns). As discussed in Section 4.2, the return number and number of returns of ground-level objects have little distinguishing power, so only intensity is considered as the LiDAR feature. We compare two feature combinations as input to RandLA-Net, which are

- **xyzRGB**: (x, y, z, R, G, B)
- **xyzRGBI**: (x, y, z, R, G, B, intensity).

5.1.1. Results on Study Area 1

Table 5.1 and 5.2 summarize the quantitative results of road type classification with RandLA-Net on Study Area 1. For both FEAT_TYPE and SURF_AREA, RandLA-Net achieves reasonable predictions. Moreover, IoU results of each road type indicate that adding intensity to the input features is beneficial to the detection of all classes.

[1] https://pytorch.org/

	Input features	OA	mA	mIoU
FEAT_TYPE	xyzRGB	83.6%	80.7%	64.1%
	xyzRGBI	**86.2%**	**82.6%**	**68.3%**
SURF_AREA	xyzRGB	88.0%	57.0%	49.9%
	xyzRGBI	**90.1%**	**61.7%**	**53.4%**

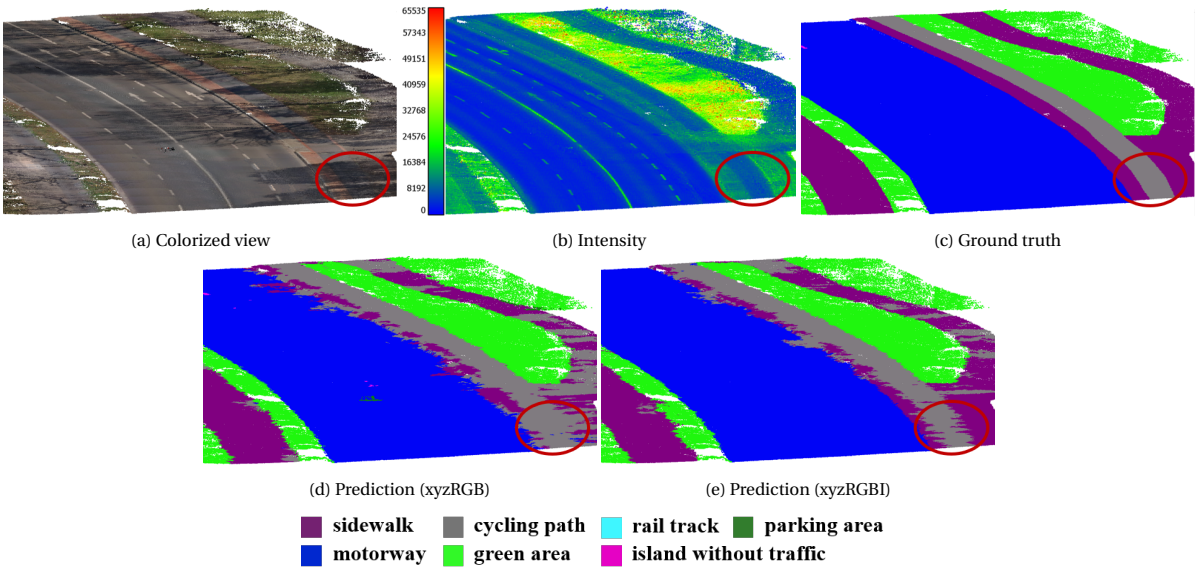
Table 5.1: Comparison of the overall accuracy (OA), mean per-class accuracy (mA), and mean Intersection over Union (mIoU) with different input features of RandLA-Net on **Study Area 1**.

Input features	sidewalk	cycling path	rail track	parking area	motorway	green area	island without traffic
xyzRGB	63.4%	48.6%	75.7%	32.8%	82.3%	83.5%	62.3%
xyzRGBI	**68.3%**	**53.5%**	**82.7%**	**36.3%**	**86.0%**	**85.5%**	**65.5%**
Input features	cobblestone	asphalt	plates	unpaved	railway		
xyzRGB	11.6%	84.0%	68.6%	85.1%	0		
xyzRGBI	**18.5%**	**86.9%**	**73.8%**	**87.8%**	0		

Table 5.2: IoU of each class with different input features of RandLA-Net on **Study Area 1**. **Top:** FEAT_TYPE (usage). **Bottom:** SURF_AREA (material).

Usage: FEAT_TYPE

Table 5.1 shows that intensity adds 4.2% to mIoU when classifying 3D road points into different FEAT_TYPE. First, point colors are easily affected by the change of illumination conditions (see Figure 5.1a), while intensity values are more stable in the case of shadows (see Figure 5.1b). Classification results in Figure 5.1d show that shadows cause confusion between *sidewalk* and *cycling path* when training with (x, y, z, R, G, B). Such confusion is largely reduced in Figure 5.1e, when the intensity feature is also considered.



(a) Colorized view (b) Intensity (c) Ground truth

(d) Prediction (xyzRGB) (e) Prediction (xyzRGBI)

Figure 5.1: Comparison of **FEAT_TYPE** classification results with features (x, y, z, R, G, B) and (x, y, z, R, G, B, intensity) on **Study Area 1**. Circled areas show the positive impact of intensity.

Also, only relying on RGB features is not enough to detect some classes. As shown in the boxed areas in Figure 5.2, there exist some traffic islands that are covered with vegetation, resulting in *island without traffic* misclassified as *green area* when only RGB features are used. Additionally, some road objects contain white markings, such as *cycling path* and *island without traffic* shown in the circled areas of Figure 5.2. Since these white markings tend to have higher reflection values than the surroundings, intensity helps to acquire clear geometrical shapes of corresponding objects in the predictions, as illustrated in Figure 5.2d.

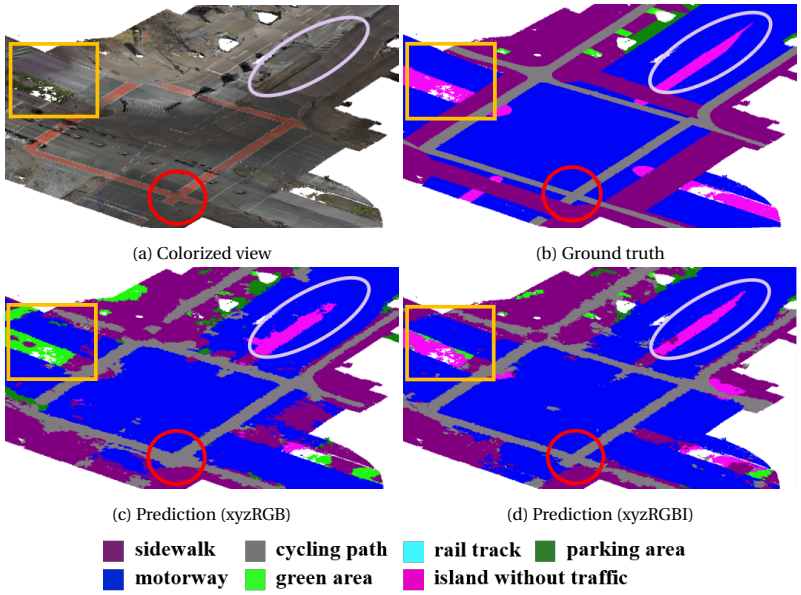


Figure 5.2: Comparison of **FEAT_TYPE** classification results using different input features of RandLA-Net on **Study Area 1**. Rectangles and circles highlight the differences between results with different feature combinations.

Material: SURF_AREA

Figure 5.3 illustrates SURF_AREA predictions using RandLA-Net. Shadows and distortions in color rendering cause incoherent RGB values in the circled areas in Figure 5.3a. As a result, classification with (x, y, z, R, G, B) shows confusion between *plates* and surrounding *asphalt* (see Figure 5.3c). Compared to colors, intensity values represent the true properties of objects in these regions, helping to achieve correct classification results (see Figure 5.3d).

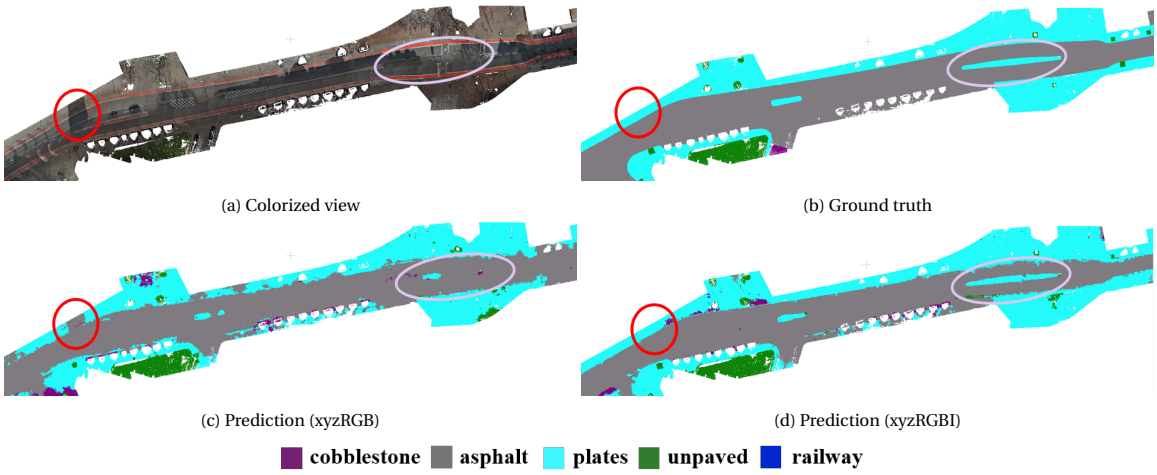
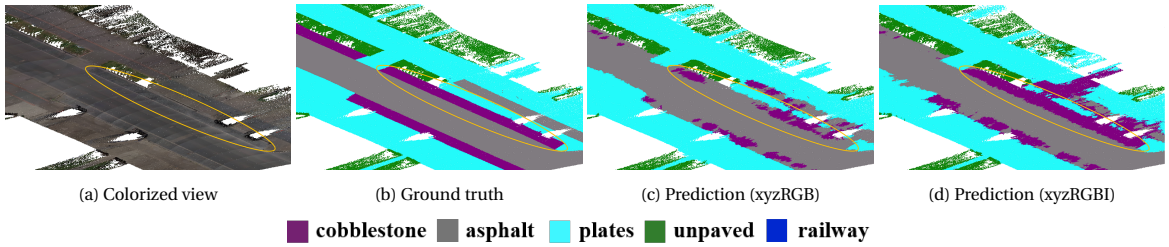


Figure 5.3: Comparison of **SURF_AREA** classification results using different input features of RandLA-Net on **Study Area 1**, with circles highlighting the impact of intensity in areas with incoherent colors.

Among five types of SURF_AREA, the classification of *railway* and *cobblestone* have the worst results, as shown in Table 5.2. First, there are very few *railway* points in Hannover since *rail track* in FEAT_TYPE is labeled as *asphalt* in SURF_AREA, which is a mistake in the ground truth annotations. As for *cobblestone*, it is hard for RandLA-Net to distinguish it from *asphalt* and *plates*. Adding intensity information mitigates this problem to some extent (see Figure 5.4), but still cannot achieve very good detection of *cobblestone*.



(a) Colorized view (b) Ground truth (c) Prediction (xyzRGB) (d) Prediction (xyzRGBI)

Figure 5.4: Comparison of **SURF_AREA** classification results using different input features of RandLA-Net on **Study Area 1**, with circles highlighting the classification of *cobblestone*.

5.1.2. Results on Study Area 2

Accuracy and IoU results of road type classification on Study Area 2 are shown in Table 5.3 and 5.4. Compared to Study Area 1, mIoU for FEAT_TYPE of Study Area 2 are lower by about 20% because of much higher variability in each class. Also, adding intensity into the input features brings similar results of overall accuracy and mIoU as only using color features.

	Input features	OA	mA	mIoU
FEAT_TYPE	xyzRGB	**76.3%**	57.2%	43.4%
	xyzRGBI	76.1%	**59.1%**	**43.7%**
SURF_AREA	xyzRGB	**84.2%**	55.4%	48.1%
	xyzRGBI	83.4%	**56.7%**	**48.7%**

Table 5.3: Comparison of the overall accuracy (OA), mean per-class accuracy (mA), and mean Intersection over Union (mIoU) with different input features of RandLA-Net on **Study Area 2**.

Input features	sidewalk	cycling path	rail track	parking area	motorway	green area	island without traffic
xyzRGB	**55.1%**	**30.4%**	**2.8%**	27.0%	78.4%	77.5%	32.8%
xyzRGBI	52.6%	30.1%	0	**27.1%**	**79.8%**	**79.5%**	**38.7%**
Input features	cobblestone	asphalt	plates	unpaved	railway		
xyzRGB	14.9%	**77.4%**	**70.3%**	77.9%	0		
xyzRGBI	**19.0%**	74.6%	68.6%	**81.2%**	0		

Table 5.4: IoU of each class with different input features of RandLA-Net on **Study Area 2**. **Top:** FEAT_TYPE (usage). **Bottom:** SURF_AREA (material).

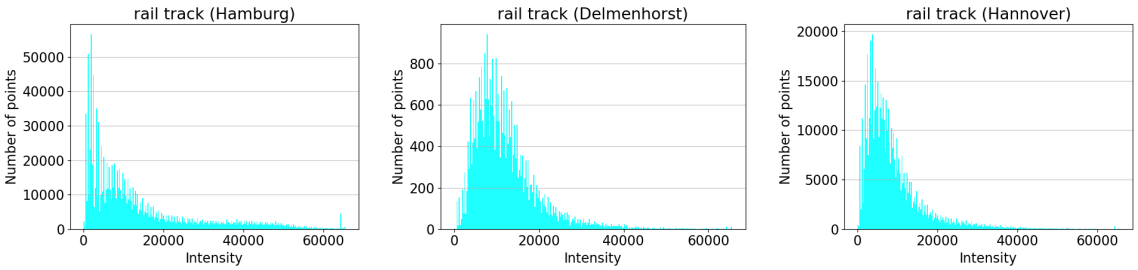
		Prediction						
		sidewalk	cycling path	rail track	parking area	motorway	green area	island
Ground Truth	sidewalk	0.69	0.11	0	0.09	0.06	0.05	0
	cycling path	0.30	0.62	0	0.02	0.04	0.01	0.01
	rail track	**0.45**	0.06	0.03	0.08	0.13	**0.21**	0.03
	parking area	0.34	0.02	0	0.51	0.10	0.03	0
	motorway	0.06	0.02	0	0.06	0.85	0	0.01
	green area	0.08	0.01	0	0.01	0	0.89	0.01
	island	0.16	0.05	0	0.01	0.16	0.20	0.42

Table 5.5: Confusion matrix of **FEAT_TYPE** classification results on **Study Area 2**, with (x, y, z, R, G, B). Elements are normalized by the total number of ground truth points in each class. Island: *island without traffic*.

Usage: FEAT_TYPE

As shown in Table 5.4, IoU values of *rail track* are much lower in contrast to training on the Hannover dataset alone (see Table 5.2). *Rail track* exists in point clouds from Hamburg, Delmenhorst, and Hannover. Hamburg

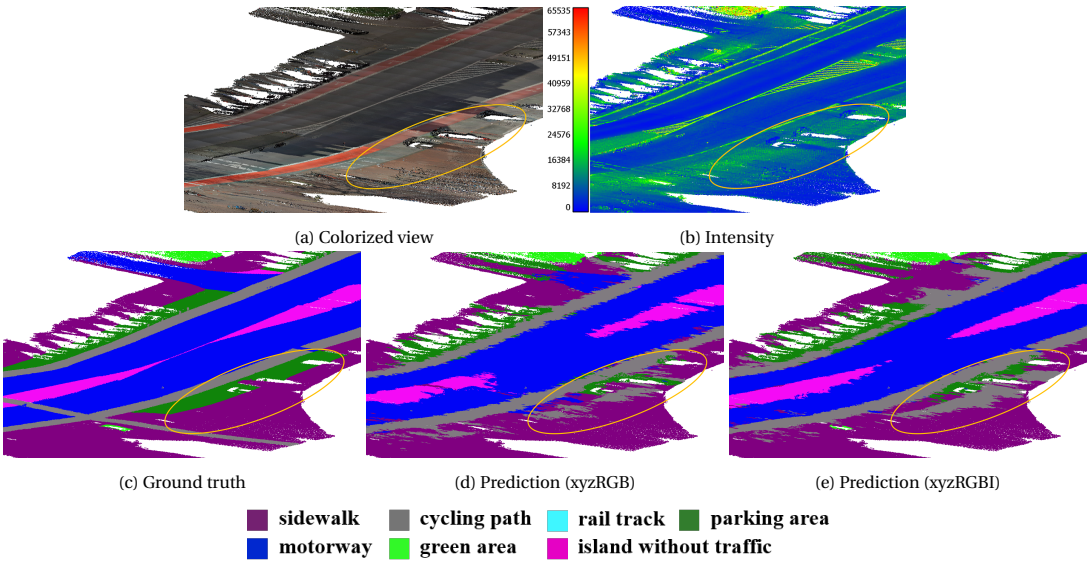
and Delmenhorst are used for training, while Hannover is adopted for testing. Although *rail track* from Delmenhorst and Hannover look similar, it shows a completely different appearance in Hamburg, as discussed in Section 4.4. The normalized confusion matrix in Table 5.5 shows that *rail track* points are mostly predicted as *sidewalk* and *green area*, which also have small elevation differences from the surrounding environment. Figure 5.5 illustrates the intensity values of *rail track* in Hamburg, Delmenhorst, and Hannover. We notice that *rail track* points from Hamburg are dominant and have more varied intensity values. As a result, the network has difficulties in generalizing to data from Hannover.



(a) Hamburg (b) Delmenhorst (c) Hannover

Figure 5.5: Histograms of *rail track* intensity in Hamburg, Delmenhorst, and Hannover.

For Study Area 2, adding intensity has a different influence compared to road type classification on Study Area 1. The classification performance of *island without traffic* is also improved, due to the significant intensity values of this road type, as shown in Figure 5.6b. On the other hand, more *sidewalk* points are classified as *cycling path* with the feature combination (x, y, z, R, G, B, intensity), causing a drop of 2.5% in the IoU result of *sidewalk*. Circled areas in Figure 5.6 refer to road segments of *sidewalk* and *cycling path* on the same surface. The similarity in intensity values of both types brings more confusion in the classification.



(a) Colorized view (b) Intensity

(c) Ground truth (d) Prediction (xyzRGB) (e) Prediction (xyzRGBI)

Figure 5.6: Comparison of **FEAT_TYPE** classification results with (x, y, z, R, G, B) and (x, y, z, R, G, B, intensity) on **Study Area 2**. Circle areas highlight the impact of the intensity on the confusion between *sidewalk* and *cycling path*.

Material: SURF_AREA

Similar to FEAT_TYPE, intensity features do not bring large improvement to the overall classification performance of SURF_AREA, as shown in Table 5.3. The increase in mA and mIoU when using (x, y, z, R, G, B, intensity) is attributed to the performance gain in *cobblestone* and *unpaved* that have unique intensity features. However, as shown in Table 5.4, IoU of both *asphalt* and *plates* drop a lot (approximately 2%) when adding intensity into input features. Since these two classes hold the most number of points among all road

types, the overall accuracy is lower than only using (x, y, z, R, G, B).

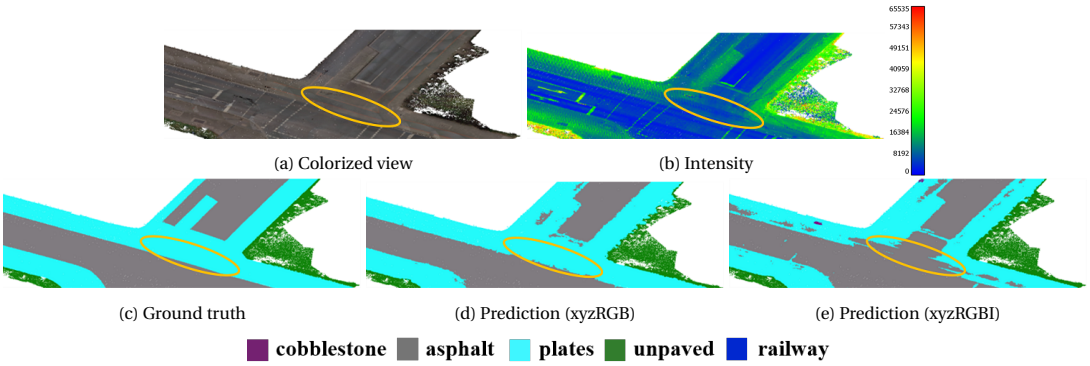


Figure 5.7: Comparison of **SURF_AREA** classification results with (x, y, z, R, G, B) and (x, y, z, R, G, B, intensity) on **Study Area 2**. Circle areas highlight a different kind of *plates* compared to the surroundings.

The circled areas in Figure 5.7e show the confusion between *asphalt* and *plates* caused by intensity features. It can be noticed that there exist two kinds of *plates* in Figure 5.7. *Plates* in the circled areas have lower intensity values, which is similar to *asphalt*. Figure 5.8 illustrates another example of confusion between *asphalt* and *plates*. With (x, y, z, R, G, B) or (x, y, z, R, G, B, intensity), we observe that *asphalt* painted in red is detected as *plates*. When using intensity, such wrong classification is especially severe, since "red" *asphalt* has much higher intensity values than the normal *asphalt*, as shown in Figure 5.8b. Although some white road markings on the *asphalt* also have strong reflections, they are not misclassified as *plates* because they are thin. Figure 5.8e also shows a wrong detection of *cobblestone*, which is caused by moving objects recorded in the point cloud. After pre-processing, the remaining points of moving objects on the ground show a distinct appearance and slightly higher intensity values compared to the surroundings. In general, the classification results of *asphalt* and *plates* in Study Area 2 are much worse than that in Study Area 1, which might be explained by the increased variability within one road type brought by data from 5 different cities.

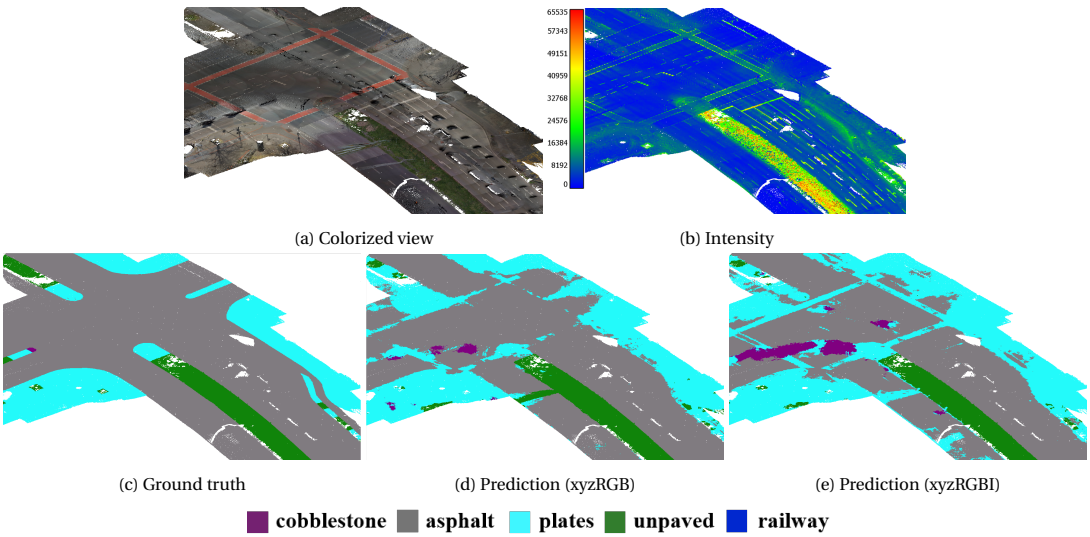


Figure 5.8: Comparison of **SURF_AREA** classification results with (x, y, z, R, G, B) and (x, y, z, R, G, B, intensity) on **Study Area 2**. Red paintings on *asphalt* can be misclassified as *plates*, especially when intensity features are added.

Nevertheless, adding intensity into input features of RandLA-Net is still beneficial to reducing the impact of shadows on road type classification. As shown in the circled area of Figure 5.9a, there are shadows caused by a car. The shadows reduce the contrast in colors between *asphalt* and *plates*, but do not affect intensity features (see Figure 5.9b). As a result, adopting (x, y, z, R, G, B, intensity) instead of (x, y, z, R, G, B) helps to

distinguish different road types, as indicated by the circled areas in Figure 5.9d and 5.9e.

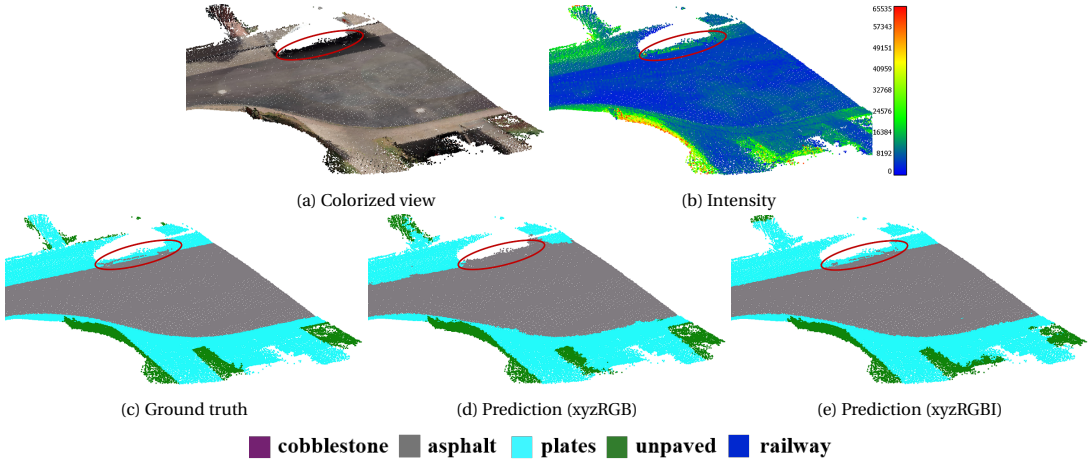


Figure 5.9: Comparison of road type (**SURF_AREA**) classification results with features (x, y, z, R, G, B) and (x, y, z, R, G, B, intensity) on **Study Area 2**. Circle areas highlight the impact of the intensity on the confusion caused by the shadow of a car.

In addition, fuzzy boundaries in road type classification can be noticed in both Study Area 1 and 2. Especially for FEAT_TYPE, inaccurate object edges are more frequently seen in the classification results than for SURF_AREA, which is caused by confusion between two road functions that are not well distinguishable in colors or intensity. In the case of SURF_AREA, such ambiguous delineation issue is less severe, since different road surface materials are more distinct in input features of the network. However, confusion between some classes in SURF_AREA still exists due to the complex road environment in urban cities. To achieve better delineation in road type classification of MLS point clouds, we need to reduce such confusion by applying certain constraints.

5.2. Effect of Embedding CRF-RNN in RandLA-Net

To improve the classification performance of RandLA-Net near boundaries, one approach in this study is to embed Conditional Random Field (CRF) as an RNN in the network. In the experiments, we set 5 iterations inside CRF-RNN. Also, RandLA-Net and the CRF-RNN module are trained end-to-end.

	CRF-RNN	OA	mA	mIoU
FEAT_TYPE	✗	83.6%	80.7%	64.1%
	✓	**83.9%**	**82.6%**	**64.9%**
SURF_AREA	✗	88.0%	57.0%	49.9%
	✓	**88.1%**	**59.1%**	**51.1%**

Table 5.6: Comparison of the overall accuracy (OA), mean per-class accuracy (mA), and mean Intersection over Union (mIoU) of road type classification on **Study Area 1** with (✓) and without (✗) CRF-RNN embedded in RandLA-Net. The input features are (x, y, z, R, G, B).

CRF-RNN	sidewalk	cycling path	rail track	parking area	motorway	green area	island without traffic
✗	63.4%	48.6%	**75.7%**	32.8%	**82.3%**	83.5%	62.3%
✓	**64.4%**	**49.1%**	73.1%	**34.3%**	82.0%	**84.5%**	**66.7%**
CRF-RNN	cobblestone	asphalt	plates	unpaved	railway		
✗	11.6%	84.0%	**68.6%**	**85.1%**	0		
✓	**18.1%**	**84.1%**	68.3%	84.9%	0		

Table 5.7: IoU of each class with (✓) and without (✗) CRF-RNN in RandLA-Net on **Study Area 1**. The input features are (x, y, z, R, G, B). **Top:** FEAT_TYPE (usage). **Bottom:** SURF_AREA (material.)

Table 5.6 and 5.7 summarize the quantitative results of road type classification on **Study Area 1** when com-

bining RandLA-Net and CRF-RNN. Using input features (x, y, z, R, G, B), CRF brings an increase of the overall performance. However, this approach is not beneficial to each class. Specifically, the IoU of *rail track* in FEAT_TYPE drops by 2.6% when adding CRF-RNN. Since CRF refines the output of RandLA-Net by strengthening the contrast between objects with different input features (e.g., coordinates and appearance), it enhances the delineation between road objects with curbs or apparent color difference. As illustrated in Figure 5.10d, the classification of *sidewalk* is largely improved by CRF-RNN, resulting in a clearer boundary between *sidewalk* and *cycling path*.

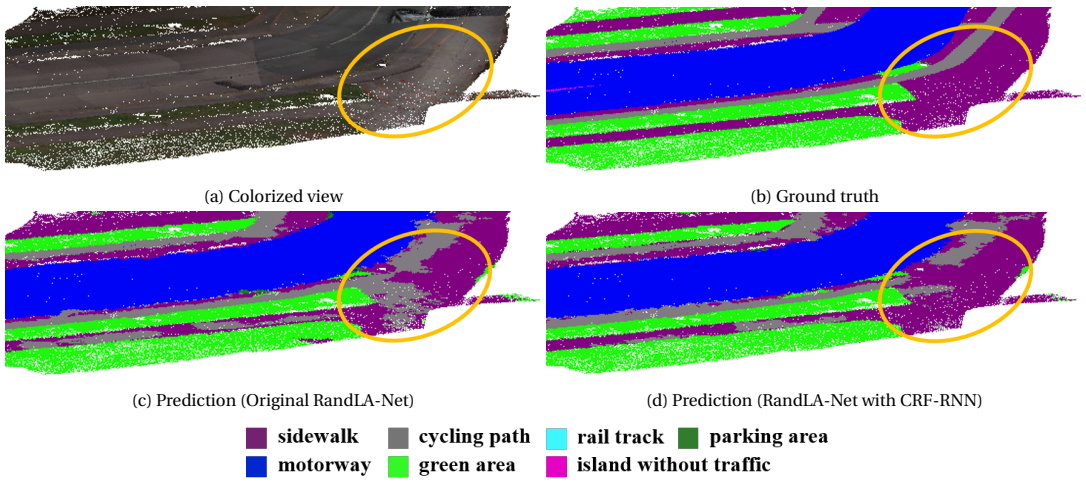


Figure 5.10: Illustration of the effect of CRF-RNN in RandLA-Net **FEAT_TYPE** classification results in **Study Area 1**, with input features (x, y, z, R, G, B). Circled areas hight the impact of the CRF-RNN module on the delineation between *sidewalk* and *cycling path*.

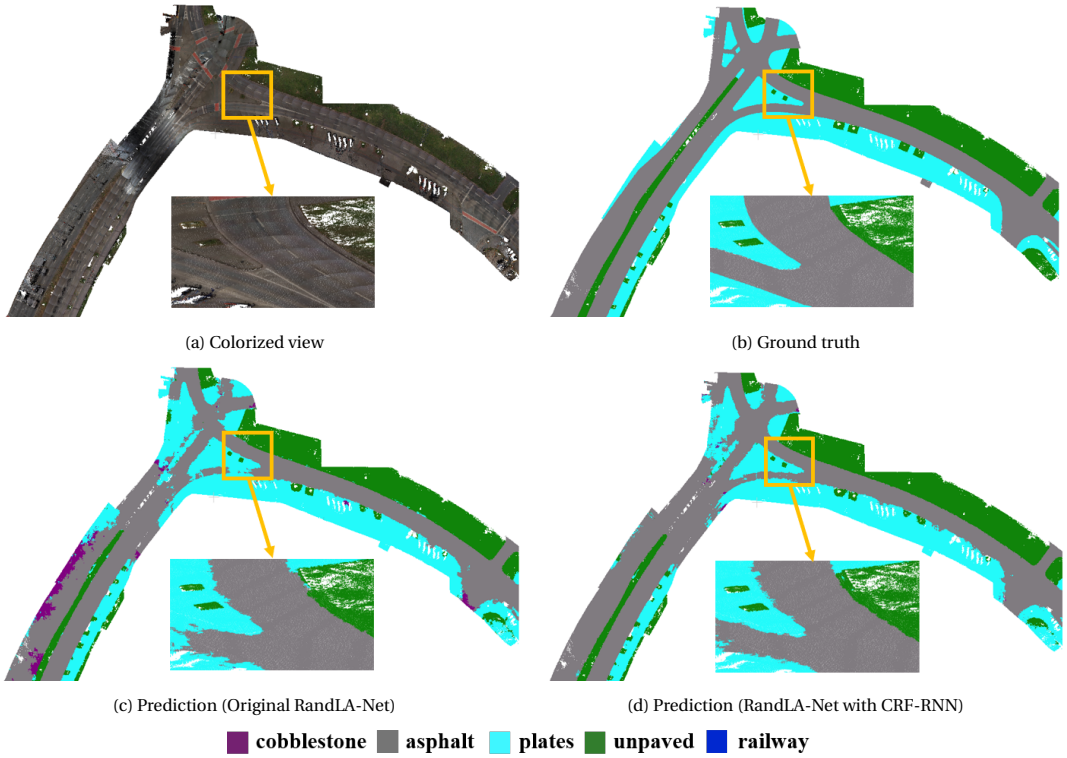


Figure 5.11: Illustration of the effect of CRF-RNN in RandLA-Net on road type (**SURF_AREA**) classification results in **Study Area 1**, with input features (x, y, z, R, G, B).

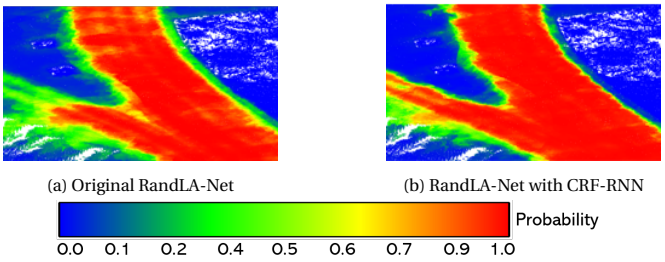


Figure 5.12: Probability maps of *asphalt* predictions from the zoomed-in areas of Figure 5.11, with input features (x, y, z, R, G, B).

In the case of SURF_AREA, the refinement effect of CRF-RNN can be observed in the classification of *cobblestone* and *asphalt*, as indicated by Table 5.7. Incorrect detection of *cobblestone* using the original RandLA-Net (see Figure 5.11c) is removed by CRF-RNN. In regard to *asphalt*, confusion in classification near the boundaries is reduced when there exists a strong contrast between objects, as shown in the boxed area in Figure 5.11d. To better illustrate the effect of CRF-RNN, we further generate probability maps of *asphalt* in the zoomed-in areas of Figure 5.11c and 5.11d by applying a *softmax* function to prediction scores of *asphalt*. In Figure 5.12, we observe higher probability values of *asphalt* near boundaries after embedding the CRF-RNN module in RandLA-Net, which demonstrates that CRF brings more "confident" delineation in the classification results.

Figure 5.13 shows a failed case of using CRF-RNN to refine *rail track* (FEAT_TYPE) predictions. It can be observed that *green area* and *rail track* highlighted by the circles show similar geometrical shapes as well as a low contrast in color. Embedding CRF-RNN does not alleviate the confusion between both road types, but cause more *rail track* points to be predicted as *green area*. To sum up, CRF cannot achieve effective improvement of classification near boundaries of objects which have similar geometry or appearance.

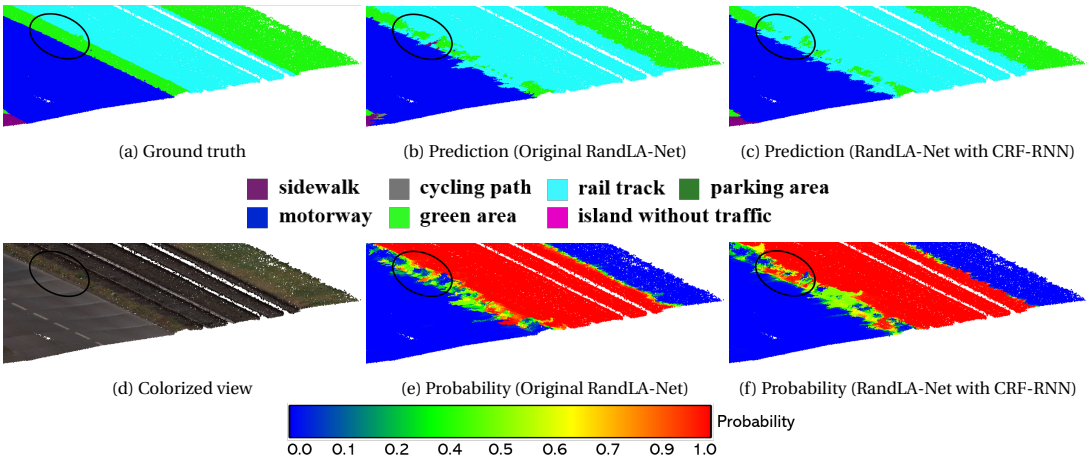


Figure 5.13: A failed case of refining **FEAT_TYPE** classification results using CRF-RNN on **Study Area 1**, with input features (x, y, z, R, G, B). **Top**: Ground truth and predicted road types. **Bottom**: Colorized point cloud and probability maps of *rail track* predictions.

5.3. Effect of Adding Distance Loss to RandLA-Net

As another strategy to improve the delineation in road type classification of MLS point clouds, multi-task learning of RandLA-Net is evaluated for both FEAT_TYPE and SURF_AREA. To achieve semantic labeling and distance prediction simultaneously, we generate discrete distance labels using a truncation threshold $R = 3.0$ m and 5 bins. The distance labels indicate the location of a point w.r.t the closest object boundary and have **6** possible values, i.e., **0.0** m, **0.6** m, **1.2** m, **1.8** m, **2.4** m, and **3.0** m.

5.3.1. Results on Study Area 1

Table 5.8 compares the overall performance of road type classification on **Study Area 1** between using single-task and multi-task learning of RandLA-Net. With (x, y, z, R, G, B), incorporating distance information has a dominant advantage in classifying FEAT_TYPE, achieving an mIoU gain of 6% compared to using the original RandLA-Net. It can also be noticed that the increase of the overall performance will not double when we use both distance loss and the feature combination (x, y, z, R, G, B, intensity). Regarding SURF_AREA, adding the distance constraint also improves the mIoU by 2.4% when using (x, y, z, R, G, B). Also, multi-task learning with input features (x, y, z, R, G, B, intensity) achieves the best results of SURF_AREA classification in all metrics, as shown in the last row in Table 5.8.

	Distance loss	Input features	OA	mA	mIoU
FEAT_TYPE	✗	xyzRGB	83.6%	80.7%	64.1%
	✓	xyzRGB	86.3%	**83.4%**	**70.1%**
	✗	xyzRGBI	86.2%	82.6%	68.3%
	✓	xyzRGBI	**86.4%**	82.2%	69.2%
SURF_AREA	✗	xyzRGB	88.0%	57.0%	49.9%
	✓	xyzRGB	89.0%	63.4%	52.3%
	✗	xyzRGBI	90.1%	61.7%	53.4%
	✓	xyzRGBI	**90.5%**	**63.6%**	**54.7%**

Table 5.8: Comparison of the overall accuracy (OA), mean per-class accuracy (mA), and mean Intersection over Union (mIoU) of road type classification on **Study Area 1** with (✓) and without (✗) the distance loss added to RandLA-Net.

Distance loss	Input features	sidewalk	cycling path	rail track	parking area	motorway	green area	island
✗	xyzRGB	63.4%	48.6%	75.7%	32.8%	82.3%	83.5%	62.3%
✓	xyzRGB	68.9%	**55.3%**	**89.8%**	**39.1%**	84.9%	85.5%	**67.1%**
✗	xyzRGBI	68.3%	53.5%	82.7%	36.3%	**86.0%**	85.5%	65.5%
✓	xyzRGBI	**69.7%**	54.9%	84.9%	36.9%	85.8%	**85.8%**	66.7%
Distance loss	Input features	cobblestone	asphalt	plates	unpaved	railway		
✗	xyzRGB	11.6%	84.0%	68.6%	85.1%	0		
✓	xyzRGB	16.7%	85.6%	72.4%	87.0%	0		
✗	xyzRGBI	18.5%	86.9%	73.8%	87.8%	0		
✓	xyzRGBI	**23.5%**	**88.1%**	**74.0%**	**88.0%**	0		

Table 5.9: IoU of each class with (✓) and without (✗) the distance loss in RandLA-Net on **Study Area 1**. Island: *island without traffic*. **Top:** FEAT_TYPE (usage). **Bottom:** SURF_AREA (material).

Usage: FEAT_TYPE

As shown in Table 5.9, detection of different road usage types benefits from the boundary information brought by distance labels differently. Generally, classification results of *cycling path*, *rail track*, *parking area*, and *island without traffic* are improved the most by multi-task learning. Figure 5.14 illustrates the effect of adding distance loss on the prediction of *cycling path*. Using (x, y, z, R, G, B), the delineation between *sidewalk* and *cycling path* is enhanced by incorporating boundary information (see Figure 5.14e). The corresponding probability map of *cycling path* also shows less confusion near road boundaries (see Figure 5.14f). However, when adding the distance loss, the circled region of *cycling path* in Figure 5.14e has false predictions. The "hole" filled with *sidewalk* predictions might be caused by the difference in distance labels between the middle part and the boundary of objects.

The circled area in Figure 5.15d shows confusion between *rail track* and *green area* when using the original implementation of RandLA-Net with features (x, y, z, R, G, B). By contrast, multi-task learning with distance loss largely reduces the confusion and improves better classification results near the boundary (see Figure 5.15e). Adding intensity to multi-task learning does not bring further improvement, as illustrated by the road

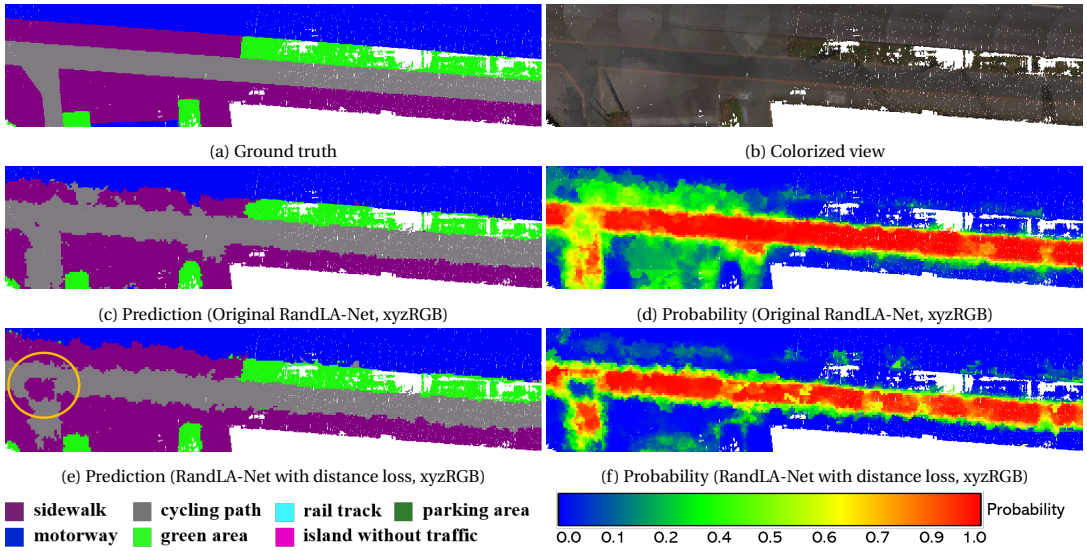


Figure 5.14: Comparisons of *cycling path* predictions with and without the distance loss added on **Study Area 1**. **Left**: Ground truth and predicted road types. **Right**: Colorized point cloud and probability maps of *cycling path* corresponding to predictions on the left.

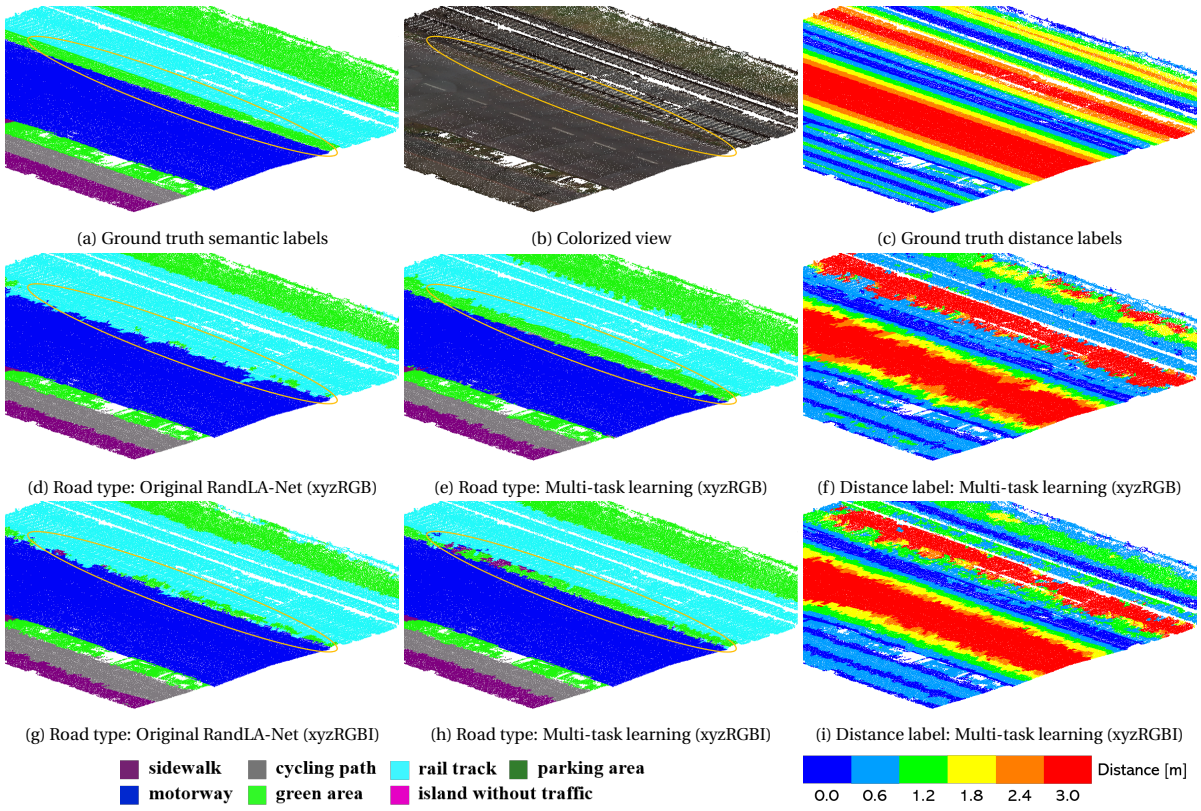


Figure 5.15: Comparisons of *rail track* predictions with and without the distance loss added on **Study Area 1**. Circled areas highlight the performance near the boundary between *rail track* and *green area*.

type classification results in Figure 5.15h and distance predictions in Figure 5.15i.

In the case of *motorway*, intensity plays a significant role in classification, as shown in Table 5.9. Using the original RandLA-Net with (x, y, z, R, G, B, intensity) achieves the best IoU of 86.0% for *motorway*. Compared to other road types, *motorway* has the lowest intensity values, making it advantageous to use (x, y, z, R, G, B, intensity) for classification. On the other hand, classification of *sidewalk* and *green area* performs the best when adopting multi-task learning and the feature combination (x, y, z, R, G, B, intensity) simultaneously.

Due to the detailed function division in urban cities, it is difficult for each road type (FEAT_TYPE) to have distinct RGB and intensity values. In general, using multi-task learning with (x, y, z, R, G, B) as input helps to achieve the best overall performance in road usage type classification on **Study Area 1**.

Material: SURF_AREA
Using multi-task learning with input features (x, y, z, R, G, B, intensity) achieves the best IoU for all road types in SURF_AREA, as shown in Table 5.9. Adding intensity into network training can bring significant information to help distinguish different road types in SURF_AREA. Moreover, boundary information can further improve the classification results of RandLA-Net with (x, y, z, R, G, B, intensity) as input, verifying the effectiveness of multi-task learning. Among all classes in SURF_AREA, the largest increase of IoU (5%) is observed in *cobblestone* when adding distance loss, no matter which input feature combination is used. As shown in Figure 5.16, with (x, y, z, R, G, B) or (x, y, z, R, G, B, intensity), only a small amount of cobblestones can be detected by the original RandLA-Net. Through adopting multi-task learning, classification of *cobblestone* is strongly improved.

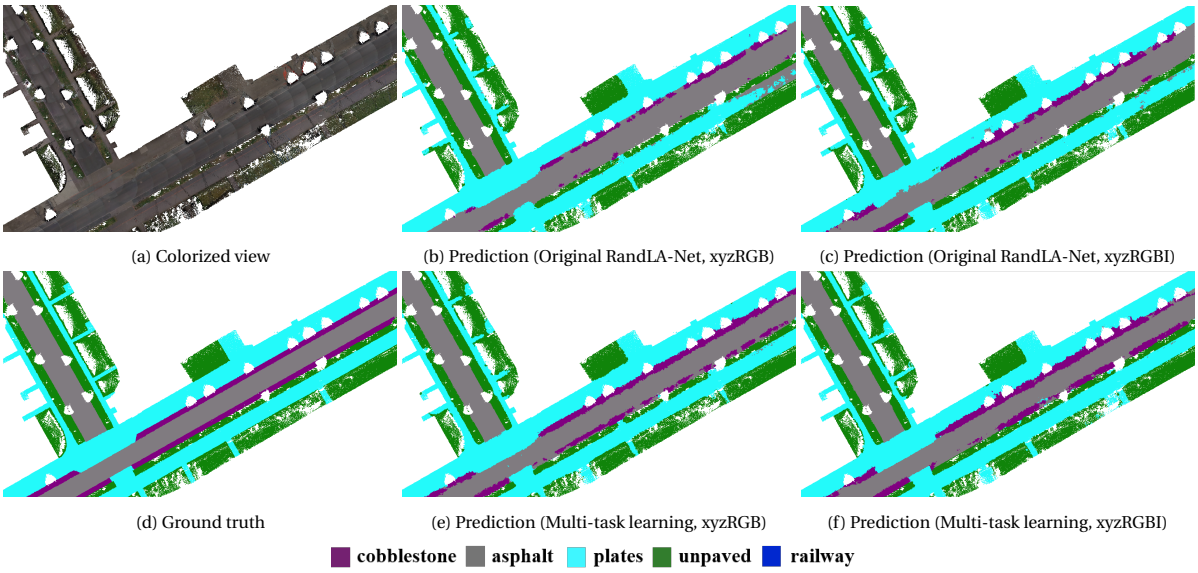


Figure 5.16: Comparisons of *cobblestone* predictions with and without the distance loss added on **Study Area 1**.

5.3.2. Results on Study Area 2

Table 5.10 and 5.11 show the positive impact of using multi-task learning on road type classification in **Study Area 2** with quantitative results. Apparently, the boundary constraint brought by the distance loss is also effective for improving the results of distinguishing road types (i.e., both FEAT_TYPE and SURF_AREA) in a larger dataset. Using the multi-task learning strategy, adding intensity into input features of the network appears to further play an important role in the classification performance.

Usage: FEAT_TYPE
Considering the results shown in Table 5.10, incorporating both distance loss and (x, y, z, R, G, B, intensity) achieves the best overall performance when classifying FEAT_TYPE in **Study Area 2**. As discussed in Section

	Distance loss	Input features	OA	mA	mIoU
FEAT_TYPE	✗	xyzRGB	76.3%	57.2%	43.4%
	✓	xyzRGB	77.6%	**60.6%**	45.2%
	✗	xyzRGBI	76.1%	59.1%	43.7%
	✓	xyzRGBI	**78.9%**	60.5%	**46.1%**
SURF_AREA	✗	xyzRGB	84.2%	55.4%	48.1%
	✓	xyzRGB	**85.6%**	58.4%	50.5%
	✗	xyzRGBI	83.4%	56.7%	48.7%
	✓	xyzRGBI	85.6%	**61.0%**	**52.0%**

Table 5.10: Comparison of the overall accuracy (OA), mean per-class accuracy (mA), and mean Intersection over Union (mIoU) of road type classification on **Study Area 2** with (✓) and without (✗) the distance loss added to RandLA-Net.

Distance loss	Input features	sidewalk	cycling path	rail track	parking area	motorway	green area	island
✗	xyzRGB	55.1%	30.4%	**2.8%**	27.0%	78.4%	77.5%	32.8%
✓	xyzRGB	55.5%	31.2%	0	32.1%	81.2%	78.2%	38.5%
✗	xyzRGBI	52.6%	30.1%	0	27.1%	79.8%	**79.5%**	38.7%
✓	xyzRGBI	**57.9%**	**31.6%**	0	**32.7%**	**82.6%**	78.6%	**38.9%**
Distance loss	Input features	cobblestone	asphalt	plates	unpaved	railway		
✗	xyzRGB	14.9%	77.4%	70.3%	77.9%	0		
✓	xyzRGB	19.1%	**80.6%**	**72.4%**	80.1%	0		
✗	xyzRGBI	19.0%	74.6%	68.6%	81.2%	0		
✓	xyzRGBI	**26.5%**	79.6%	71.7%	**82.4%**	0		

Table 5.11: IoU of each class with (✓) and without (✗) the distance loss in RandLA-Net on **Study Area 2**. Island: *island without traffic*. **Top:** FEAT_TYPE (usage). **Bottom:** SURF_AREA (material).

5.1.2, in Study Area 2, adding intensity in training the original RandLA-Net does not bring much performance gain compared to only using RGB features. However, when using multi-task learning, adding intensity in general improves the FEAT_TYPE classification results in Study Area 2 to a large extent.

Note from Table 5.11 that multi-task training using (x, y, z, R, G, B, intensity) as input helps to obtain the best classification in all road types except for *rail track* and *green area*. *Rail track* in Hamburg shows a completely different appearance from that in Hannover and Oldenburg. Since Hamburg is used for training and has the largest coverage of *rail track*, *rail track* points in the test dataset (i.e., Hannover and Oldenburg) are seldom detected. The usage of intensity and distance labels strengthens the contrast between *rail track* in different cities. With regard to *green area*, adding distance loss improves the classification performance when using (x, y, z, R, G, B), but does not play a positive role with (x, y, z, R, G, B, intensity) as input. Classification of road types such as *cycling path* and *parking area* is improved by adding the distance loss to RandLA-Net, while it is not really affected by adding intensity into input features. Also, with (x, y, z, R, G, B) or (x, y, z, R, G, B, intensity), the performance increase brought by multi-task learning is similar, as shown in Table 5.11.

As for *sidewalk* and *motorway*, adding intensity to input features harms the classification results when using the original RandLA-Net. However, with the constraint of distance labels, using intensity further increases the IoU results of *sidewalk* and *motorway* (see Table 5.11). Figure 5.17 illustrates the effect of intensity features and boundary information on the FEAT_TYPE classification in Study Area 2. From Figure 5.17b, we notice that *sidewalk* in this region has similar reflection values as *parking area*. Training the original RandLA-Net with the feature combination (x, y, z, R, G, B, intensity) causes more confusion between *sidewalk* and *parking area*, as shown in Figure 5.17f. With (x, y, z, R, G, B), adding the distance loss to network training reduces confusion near boundaries between *sidewalk* and *motorway*. Based on multi-task learning, using intensity features enhances the delineation between *sidewalk* and *motorway* to a large extent, as shown in Figure 5.17g.

Material: SURF_AREA

Quantitative results in Table 5.10 and 5.11 indicate that the classification of SURF_AREA in Study Area 2 also benefits from multi-task learning. Similar to FEAT_TYPE, intensity features have a positive impact on road

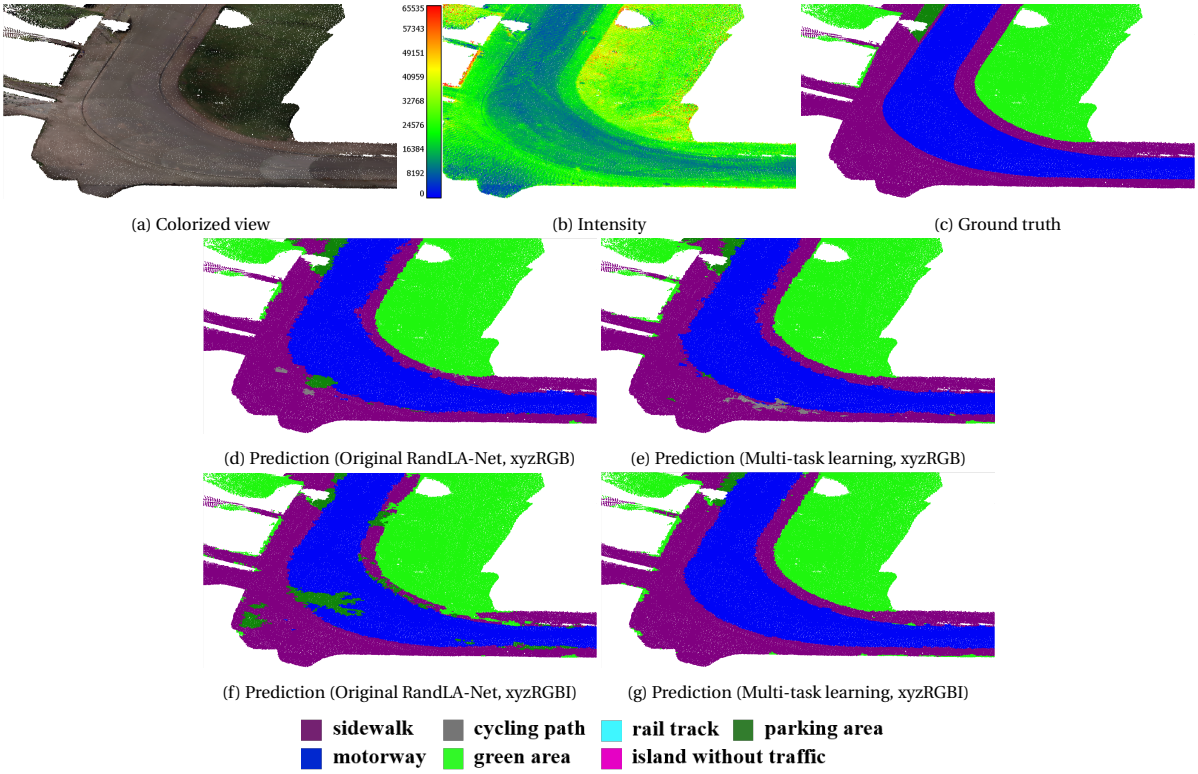


Figure 5.17: Comparison of **FEAT_TYPE** classification results on **Study Area 2**.

type classification with the constraint of distance loss. Especially for *cobblestone*, using multi-task learning in combination with input features (x, y, z, R, G, B, intensity) brings much better performance than other setups. Figure 5.18 also highlights the positive effect of multi-task learning on the classification of *asphalt* and *plates*. With (x, y, z, R, G, B) or (x, y, z, R, G, B, intensity) as input, adding boundary constraint helps to distinguish *asphalt* and *plates* that have height differences but similar appearance, resulting in a clearer delineation in point cloud classification.

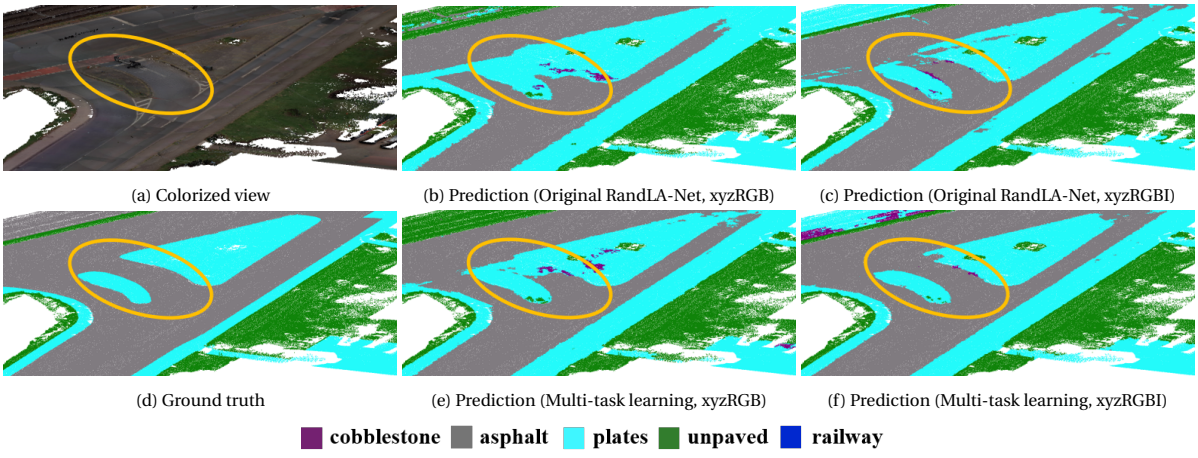


Figure 5.18: Comparison of **SURF_AREA** classification results on **Study Area 2**. Circled areas highlight the impact of multi-task learning.

In summary, the multi-task learning strategy proposed in this study is beneficial to mitigating confusion between road types having similar features. By adding the distance loss, we incorporate boundary constraints into network training, which improves the overall road type classification results as well as the delineation performance, especially on Study Area 2 that covers a larger area and has higher data variability.

6 Discussion

This chapter provides discussions about the road type classification results. Section 6.1 discusses how the classified point clouds can be applied to achieve vectorized road boundaries. Moreover, we present comparisons among road type classification with different input point densities of the network in Section 6.2.

6.1. Road Boundary Vector Extraction

To achieve a relatively complete overview of roads within one city, we extract road boundary vectors from the best point cloud classification results in the test areas of **Study Area 2**, i.e., Hannover and Oldenburg. The vectorization results for the material type SURF_AREA are also compared with that obtained by an image-based method used by Cyclomedia.

6.1.1. Results on Study Area 2

In this study, we vectorize road boundaries from classified MLS point clouds by creating polygons around clusters of three or more points, with an aggregation distance between points of 1 m to preserve the details such as some narrow paths. These polygons represent the $alpha$-shapes of classified road points. Specifically, we generate 2D polygons for each road segment using the tool *Aggregate Points* implemented by ArcGIS Pro. Moreover, overlaps between polygons are removed by the tool *Remove Overlap* with the method *Thiessen*, which divides the overlap area using a straight line, to ensure processing efficiency. Afterwards, the polygons are converted into rasters using a grid size of 0.1 m. We randomly sample 10^6 points from the raster for evaluation.

Figure 6.1 shows parts of the extracted boundary vectors for road usage type (FEAT_TYPE), which are generated from point cloud classification results acquired by multi-task learning with (x, y, z, R, G, B, intensity) that achieves the highest mIoU values (see Table 5.10). It can be seen that the achieved delineation between road objects is better for relatively simple road scenes (see Figure 6.1d). For complex scenes, as the crossroad in Figure 6.1e, it is difficult to obtain accurate road boundaries due to class confusions in the point cloud classification results. Although the distance loss in the network provides geometric constraints, it cannot apply hard constraints such as a height threshold on boundaries. As a result, perfect delineation is still not ensured in the point cloud classification, which causes less smooth boundary vectors compared to the ground truth polygons. To generate useful map products, additional processing, e.g., regularization, for the vectorization results might be required. Furthermore, narrow objects like *cycling path* have bad connectivity in the polygon results (see Figure 6.1e), which is partly due to confusion with classes such as *sidewalk* in the road type classification results. Figure 6.1e and 6.1f also illustrate a lot of "holes" in the polygon extraction results. These "holes" are mainly caused by the removal of non-road points (e.g., cars) during the pre-processing of MLS points. During laser scanning, some obstacles on the road also lead to gaps in the point cloud. It can be seen that the completeness of boundary vector extraction is largely affected by these missing data.

Table 6.1 summarizes the confusion matrices for FEAT_TYPE annotations in Hannover and Oldenburg compared to the ground truth polygons. For both cities, the recall of *cycling path* is much higher than the precision. Although a large area of the *cycling path* presented in the ground truth is detected, the extracted poly-

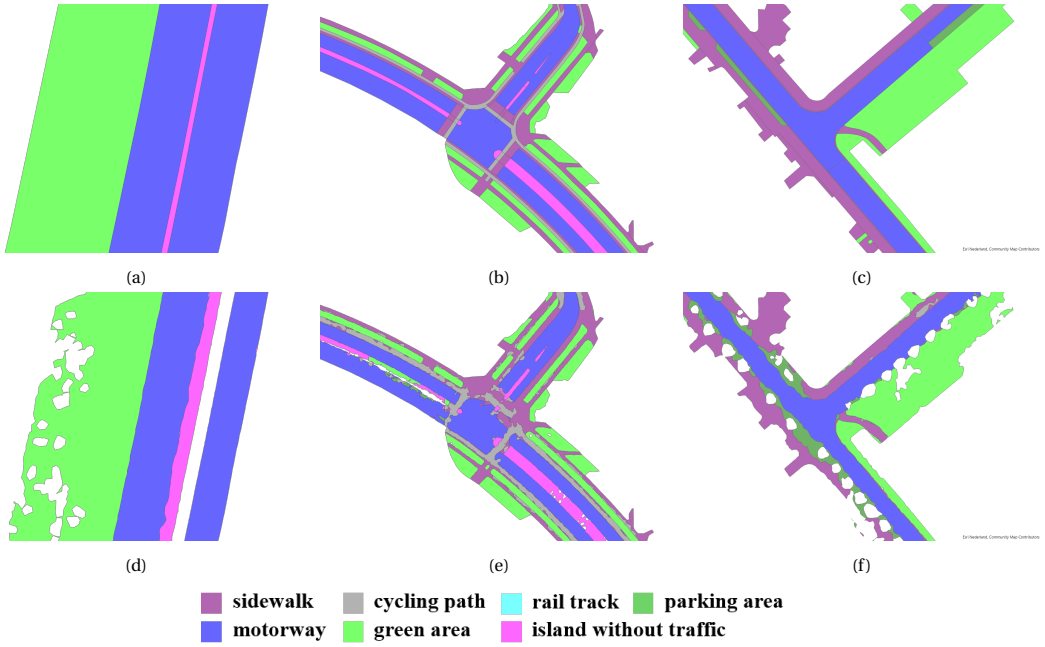


Figure 6.1: Illustration of road boundary extraction results for FEAT_TYPE on **Study Area 2**. **Top:** Ground truth. **Bottom**: Boundary polygons achieved by our method.

gons in this class are of low correctness. By contrast, *motorway* and *green area* show relatively high values in both precision and recall results, indicating good boundary extraction results.

		sidewalk	cycling path	rail track	parking area	motorway	green area	island
Hannover	**Precision**	69.2%	31.6%	nan	51.2%	94.1%	90.9%	55.8%
	Recall	67.0%	68.5%	0	62.4%	85.8%	90.1%	48.4%
Oldenburg	**Precision**	81.4%	41.0%	-	43.9%	93.3%	81.5%	56.5%
	Recall	72.7%	78.5%	-	42.4%	89.0%	91.4%	53.7%

Table 6.1: Precision and recall showing the comparison between extracted boundary polygons and the ground truth in **Study Area 2**, with the road label **FEAT_TYPE**. Boundary polygons are obtained from point cloud classification results on **Study Area 2** using multi-task learning based on RandLA-Net, with (x, y, z, R, G, B, intensity) as input. Island: *island without traffic*.

6.1.2. Comparison to Image-based Method

Cyclomedia also achieved polygon extraction of different road types based on semantic segmentation results of street-view images that were acquired at the same time as the MLS point clouds. The segmentation was performed through Mask-RCNN [16], which is a neural network designed for image semantic segmentation. Moreover, the network was also trained on **Study Area 2**, with RGB features as well as the same train-test split used in this thesis project. To evaluate the obtained polygons, the vectors were also first rasterized with a grid size of 0.1 m. Table 6.2 shows the precision and recall results of both the image-based and point cloud-based method proposed in this research in the city of Oldenburg. The point cloud classification is performed with multi-task learning based on RandLA-Net and the feature combination (x, y, z, R, G, B).

Compared to the image-based method, we obtain higher recall values for every class through road type classification of MLS point clouds, demonstrating the effectiveness of the road boundary extraction approach in this thesis project. Especially in the case of *cobblestone*, *asphalt*, and *plates*, the recall is improved by approximately 15% using the point cloud-based method. As for the precision, the image-based method achieves a much better value (73.7%) for *cobblestone*, showing that annotating *cobblestone* with point cloud classification results has lower correctness for this road type.

Figure 6.2 also illustrates the boundary polygons achieved by both image-based and point cloud-based meth-

Method		cobblestone	asphalt	plates	unpaved
Image-based	**Precision**	73.7%	64.1%	77.3%	68.6%
	Recall	17.3%	73.7%	69.1%	84.3%
Point cloud-based	**Precision**	47.6%	87.4%	85.5%	83.2%
	Recall	31.9%	89.3%	86.3%	85.5%

Table 6.2: Precision and Recall acquired by the image-based and point cloud-based method in **Oldenburg**, with the road label **SURF_AREA**. In the image-based method, boundary polygons are obtained from image semantic segmentation results on **Study Area 2** using Mask-RCNN, with RGB features. In the point cloud-based method, boundary vectors are obtained from the point cloud classification results on **Study Area 2** using multi-task learning based on RandLA-Net, with (x, y, z, R, G, B) as input.

ods. Compared to the ground truth annotations, road boundary vectors acquired from point cloud classification results are more reasonable. The strength of image-based method is that there are no holes or gaps caused by certain pre-processing steps. Although both methods use the same color features for road type classification, the point cloud-based method also takes advantage of the 3D geometry of different objects, resulting in better classification results and more accurate delineation. On the other hand, since no depth information is included when training Mask R-CNN, we also have to admit that such comparisons between solutions based on 3D point cloud classification and 2D image segmentation are relatively unfair.

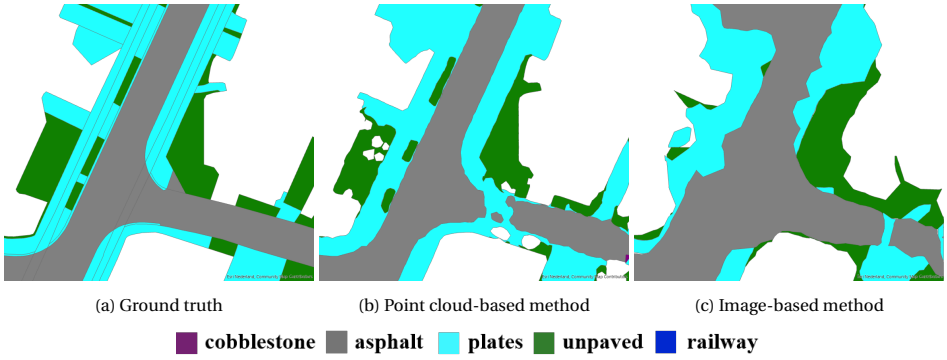


Figure 6.2: Road boundary extraction for **SURF_AREA** on **Study Area 2** from the images-based method and point cloud classification.

6.2. Investigation of Input Point Density of RandLA-Net

To select an appropriate input point density for road type classification in this study, we train the original RandLA-Net with three different point intervals, i.e., 0.1 m, 0.2 m, and 0.3 m, on **Study Area 1**. To this end, we uniformly downsample the point clouds before feeding them into RandLA-Net. Note that predictions of all set-ups will be upsampled using nearest interpolation to have the same data resolution (0.1 m). During training, we adopt (x, y, z, R, G, B) as input and $K = 16$ nearest neighbors in local feature aggregation. In addition, the experiments are conducted separately for FEAT_TYPE (usage) and SURF_AREA (material). For FEAT_TYPE, we use the original 9 classes, i.e., *sidewalk*, *cycling path*, *rail track*, *parking area*, *motorway*, *green area*, *island without traffic*, *pedestrian area*, and *others*. We observe that different point densities help to achieve the best classification results of FEAT_TYPE and SURF_AREA.

Table 6.3 presents the quantitative results of classifying FEAT_TYPE and SURF_AREA with different point densities. We achieve the best mA (68.3%) and mIoU (53.1%) with a point interval of 0.2 m when classifying FEAT_TYPE. Point density reflects how many details are captured in the point clouds. With the same number of neighboring points ($K = 16$), point interval also affects the receptive field of the network. From the circled areas in Figure 6.3, it can be noticed that a smaller receptive field (with point interval 0.1 m) is not beneficial to detecting *cycling path* that is narrow but long. The point interval of 0.3 m achieves a larger receptive field, but fewer details of the road scene with sparser points, resulting in worse connectivity of *cycling path*, as shown in Figure 6.3e.

By contrast, the classification of material types requires a small data resolution to perceive detailed structures

	Point interval (m)	OA	mA	mIoU
FEAT_TYPE	0.1	79.9%	65.4%	50.3 %
	0.2	81.0%	**68.3%**	**53.1%**
	0.3	**81.1%**	65.3%	51.7%
SURF_AREA	0.1	**88.0%**	**57.0%**	**49.9%**
	0.2	83.7%	52.6%	44.7%
	0.3	75.8%	47.8%	38.7%

Table 6.3: Comparison of the overall accuracy (OA), mean per-class accuracy (mA), and mean Intersection over Union (mIoU) with a different input point density of RandLA-Net on **Study Area 1**. The classification of FEAT_TYPE contains 9 road types.

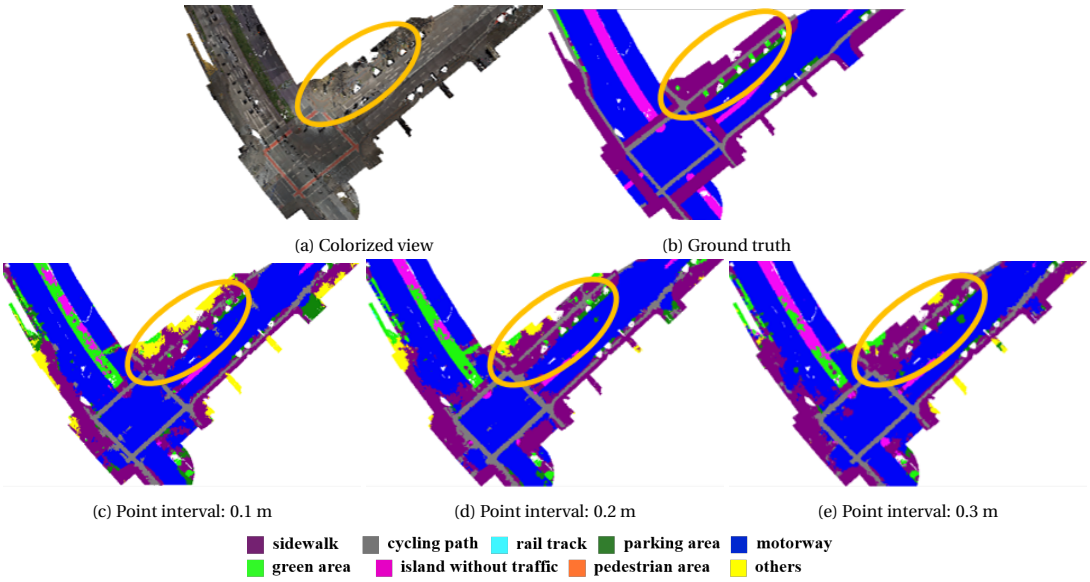


Figure 6.3: Road type classification for **FEAT_TYPE** on **Study Area 1** using RandLA-Net with different point densities.

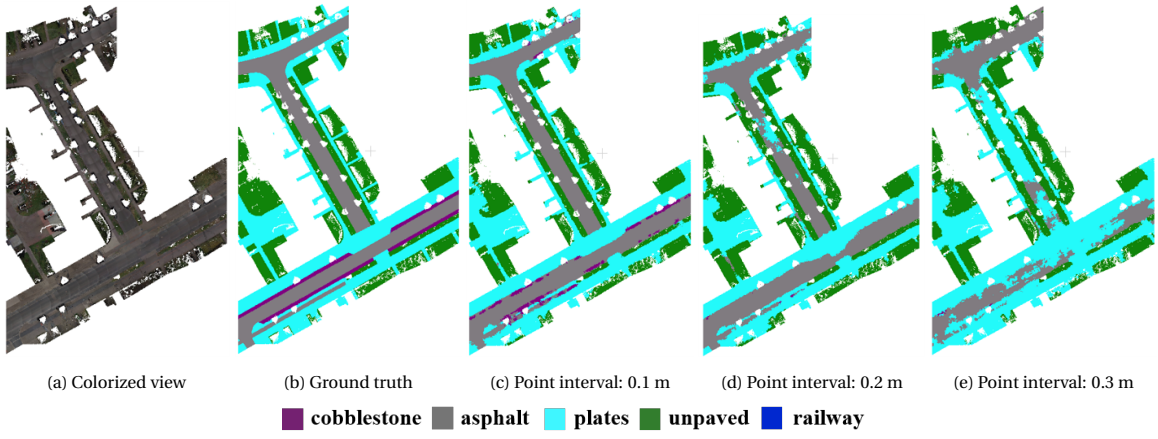


Figure 6.4: Road type classification for **SURF_AREA** on **Study Area 1** using RandLA-Net with different point densities.

of the road objects. With the point interval of 0.1 m, we obtain the highest mA (57.0%) and mIoU (49.9%) for SURF_AREA. For objects like *cobblestone* shown in Figure 6.4, they cannot be distinguished between *asphalt* or *plates* using sparser data. Indeed, using both small point intervals and more neighboring points (i.e., a larger K value) during local feature aggregation might be the optimal solution. However, considering the computational cost of using more points, we need to make the trade-off between data resolution and scope of the network into account.

7

Conclusions & Recommendations

In this chapter, we conclude this thesis project in Section 7.1 by answering the research questions raised in Chapter 1. Afterwards, Section 7.2 presents recommendations for future research on road type classification of MLS point clouds and road boundary vector extraction.

7.1. Conclusions

In Section 1.5, the main research question of this project is summarized as:

- **How to acquire road types from MLS point clouds accurately and efficiently?**

In general, we achieve road types, i.e., usage and material type, in polygon representations through point cloud classification by deep learning. To obtain the road type of each point on the road surface from MLS point clouds, we utilize the main structure of a point-wise neural network, i.e., RandLA-Net. Furthermore, to improve the classification performance near road boundaries, we encode the boundary information as distance labels and incorporate them into network training using multi-task learning. Finally, we generate road boundary polygons from the classified point clouds through post-processing. The achieved polygons indicate boundary locations of road segments in different types.

The main research question is split into 6 sub-questions, which are answered in detail as follows.

1. **What kind of pre-processing strategies should be applied to MLS point clouds?**

   In this project, we use the MLS point cloud data acquired by Cyclomedia. Considering the large volume of original point clouds, as introduced in Chapter 4, we downsample them with a uniform point spacing. To account for the noise brought by unlabeled objects (e.g., buildings and trees) and moving objects on the road, we also apply ground filtering to remove non-road points.

2. **How to realize road type classification of MLS point clouds through deep learning?**

   Road type classification of MLS point clouds is consistent with the aim of point cloud semantic segmentation, which is a crucial step for road boundary mapping in this study. As described in Chapter 3, we follow the architecture of RandLA-Net for road type classification. By exploiting such a point-wise neural network, we avoid unnecessary data format conversion (e.g., from point clouds to images or voxels). the random sampling strategy in RandLA-Net also enables to consume a large number of points at one time for processing, while the local feature aggregation module helps to extract rich local information around each point with 3D coordinates and additional point features (e.g., color and intensity) as input.

   Specifically, we apply RandLA-Net on road point clouds after ground filtering. To achieve the usage type (FEAT_TYPE) and material type (SURF_AREA) of each point, we train the network on two labels separately. We also select the suitable point density input to RandLA-Net for both labels through experiments, in order to account for the different complexity of classifying FEAT_TYPE and SURF_AREA. Apart from point density, which affects the receptive field of the neural network, input point features

determines information used to interpret the road scene. As discussed in Chapter 5, intensity is significant for improving classification performance in the case of shadows and distinguishing different road objects with similar colors.

3. **How to alleviate the fuzzy boundary issue in 3D point cloud classification?**

   Using the original implementation of RandLA-Net, we observe ambiguous point cloud classification results near road boundaries, which is partly caused by the loss of low-level information during progressive random sampling. Chapter 3 describes two strategies in this study to mitigate the fuzzy delineation issue, i.e., refining predictions of RandLA-Net by a CRF-RNN module and adding boundary constraints during training RandLA-Net. Compared to CRF-RNN, the latter method achieves better performance.

   To incorporate boundary information into network training, we construct discrete distance labels that represent 2D distances from each point to their closest road boundary. By adding another fully connected layer at the end of RandLA-Net, we train on two point cloud classification tasks, i.e., road type and distance prediction, simultaneously. Through comparisons with the original implementation of RandLA-Net shown in Chapter 5, we demonstrate the effectiveness of the multi-task learning strategy on both Study Area 1 and 2. However, adding soft constraints in the deep learning network only reduces class confusion near road boundaries, but still cannot ensure smooth delineation in point cloud classification results.

4. **How is the generalization ability of the proposed method regarding different point cloud datasets?**

   Chapter 4 introduces two study areas used in this project, i.e., Study Area 1 covering only Hannover and Study Area 2 consisting of 5 different German cities. As shown in Chapter 5, compared to Study Area 1, experiments on Study Area 2, i.e., the larger dataset, have slightly lower quantitative results for material type SURF_AREA, but much worse classification results for usage label FEAT_TYPE. We notice that the variability of each road usage type among different cities affects the generalization ability of the network. For instance, when testing on Hannover and Oldenburg, the model fails to classify *rail track* points that are different from those in the training dataset, i.e., Hamburg, Delmenhorst, and Bremerhaven. To sum up, the generalization ability of our proposed method can still be improved to account for datasets with high variability in each class.

5. **Given the road type classification results, how to extract the road boundary vectors effectively?**

   We describe the post-processing steps adopted in this study to generate road boundary vectors in Chapter 3. Given the road type of each point acquired from point cloud classification, 2D polygons that closely fit the geometrical shape of each classified road segment are desired as the final output. We achieve vectorized road boundaries through creating polygons around clusters of three or more points, as implemented in ArcGIS Pro. We admit that the quality of vector products is dependent on point cloud classification results in the previous step. As a result, false predictions in point cloud classification cannot be corrected through post-processing in the proposed method and are still presented in the extracted boundary polygons.

6. **How good are the road boundaries achieved by our method compared to other methods, e.g., image-based approaches?**

   In Chapter 6, We compare road boundary polygons obtained by our method with an approach based on image semantic segmentation by Mask R-CNN used by Cyclomedia. We conclude that our method achieves better road boundaries overall through both quantitative and qualitative evaluation. Specifically, polygons acquired by point cloud classification represent much more accurate and regularized road boundaries. Also, some small road objects, e.g., *green area* scattered on the roadside, are well presented in the boundary polygons achieved by our method.

In summary, we incorporate boundary constraints into a point-wise neural network to achieve automatic road type classification from large-scale MLS point clouds in dense urban areas. Through the multi-task training strategy, boundary and semantic information of the road are learned simultaneously and helping each other in optimization, avoiding additional refinement of road type predictions. Improved classification performance near boundaries also ensures the quality of road boundary vector products in the end.

7.2. Recommendations

Regarding the proposed pipeline for 3D road boundary mapping in this thesis project, there are several aspects that can be further investigated in future research, as described below.

- **Better utilization of distance labels**

  Distance labels in this study measure the distance from each point to its corresponding road boundary. Notably, distance labels equal to 0 indicate the boundary locations in the point clouds. Currently, the predicted distance labels are only used to promote the classification of semantic labels, but not further helping to extract boundary vectors since distance predictions achieved in this project cannot represent continuous road boundaries. Nevertheless, the encoding of distance labels provides a way of segmenting boundary points directly in future study.

- **Weighting strategy in multi-task learning**

  We adopt equal weighting of road type classification loss and distance loss in multi-task learning. Although the proposed method brings significant improvement in point cloud classification, it is valuable to explore the effect of assigning different importance to two loss functions.

- **Improvement of extracted road boundary vectors**

  As a post-processing step, road boundary vector extraction in this study is highly affected by the point cloud classification results. Failure in classifying some narrow paths results in bad connectivity of objects like *cycling path* in the final road boundary vectors. Considering the topology of the road network, the disconnectivity issue can be further mitigated. Moreover, to achieve usable road mapping products, we need to fill the "holes" in boundary polygons caused by the ground filtering step and simplify the shape of some boundary vectors. Also, the current way of solving overlapping polygons is simple but brutal. Better strategies to remove overlap can be considered, e.g., taking the labeling priority of different road types into account.

- **Comparisons with other methods**

  For road type classification of MLS point clouds, we compare the performance of RandLA-Net with other neural networks, i.e., PointCNN and GACNet, in Appendix A. These networks aggregate local information at the point level. Performance of methods like superpoint graph network [22], which utilizes the idea of over-segmentation, can also be investigated in the future.

  For road boundary vectors, we compare the quality of polygon products acquired from point cloud-based and image-based methods. Since image semantic segmentation used by Cyclomedia does not include depth information, such comparison between 3D and 2D solutions is relatively unfair. In future research, comparisons with other 3D or 2.5D approaches can be carried out.

- **Generalization to complex datasets**

  Road usage type (FEAT_TYPE) classification results on Study Area 1 and 2 indicate that the generalization ability of the proposed method on complex datasets can still be improved. To achieve better generalization performance, we might need to modify the network architecture or finetune the model with a small amount of data in the testing area.

- **Adaptive downsampling**

  During the pre-processing of MLS point clouds, we use uniform downsampling to reduce the data volume, which treats points on road surfaces and boundaries equally. To put more focus on the classification performance near boundaries and avoid unnecessary computation for road surface points, more advanced downsampling strategies, e.g., achieving adaptive point density for different road components, can be further investigated.

- **Simultaneous classification of FEAT_TYPE & SURF_AREA**

  Through experiments we demonstrate that different point densities suit the classification of road usage type (FEAT_TYPE) and material type (SURF_AREA). Also, some class definitions in both road types are contradictory, e.g., *rail track* in FEAT_TYPE and *railway* in SURF_AREA. Therefore, we train on FEAT_TYPE and SURF_AREA separately in this thesis project. However, FEAT_TYPE can also be related to SURF_AREA to a large extent. For example, *green area* in road usage corresponds to *unpaved* in

material. Moreover, *motorway* is always made of *asphalt*. To better utilize the relevance between road usage and material type, simultaneous learning of both labels is worth investigation in the future.

Bibliography

[1] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5221–5229, 2017.

[2] J Balado, Lucía Díaz-Vilariño, P Arias, and Higinio González-Jorge. Automatic classification of urban ground elements from mobile laser scanning data. *Automation in Construction*, 86:226–239, 2018.

[3] Dana H Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2): 111–122, 1981.

[4] Benjamin Bischke, Patrick Helber, Joachim Folz, Damian Borth, and Andreas Dengel. Multi-task learning for segmentation of building footprints with deep neural networks. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1480–1484. IEEE, 2019.

[5] Gunilla Borgefors. Distance transformations in digital images. *Computer vision, graphics, and image processing*, 34(3):344–371, 1986.

[6] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[7] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

[8] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[9] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[11] David Griffiths and Jan Boehm. A review on deep learning techniques for 3D sensed data classification. *Remote Sensing*, 11(12):1499, 2019.

[12] Eleonora Grilli, Fabio Menna, and Fabio Remondino. A review of point clouds segmentation and classification algorithms. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42:339, 2017.

[13] Haiyan Guan, Jonathan Li, Yongtao Yu, Michael Chapman, and Cheng Wang. Automated road information extraction from mobile laser scanning data. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):194–205, 2014.

[14] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3D point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[15] Zeeshan Hayder, Xuming He, and Mathieu Salzmann. Boundary-aware instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5696–5704, 2017.

[16] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[17] Qing Hou and Chengbo Ai. A network-level sidewalk inventory method using mobile LiDAR and deep learning. *Transportation research part C: emerging technologies*, 119:102772, 2020.

[18] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[19] Arshad Husain and Rakesh Chandra Vaishya. Road surface and its center line and boundary lines detection using terrestrial LiDAR data. *The Egyptian Journal of Remote Sensing and Space Science*, 21(3): 363–374, 2018.

[20] Seung-Hun Kim, Chi-Won Roh, Sung-Chul Kang, and Min-Yong Park. Outdoor navigation of a mobile robot using differential GPS and curb detection. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3414–3419. IEEE, 2007.

[21] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with gaussian edge potentials. *Advances in neural information processing systems*, 24:109–117, 2011.

[22] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with Superpoint Graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[23] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution on X-transformed points. *Advances in neural information processing systems*, 31:820–830, 2018.

[24] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019.

[25] Ze Liu, Han Hu, Yue Cao, Zheng Zhang, and Xin Tong. A closer look at local aggregation operators in point cloud analysis. *arXiv preprint arXiv:2007.01294*, 2020.

[26] Xiaoxin Mi, Bisheng Yang, Zhen Dong, Chi Chen, and Jianxiang Gu. Automated 3D road boundary extraction and vectorization using MLS point clouds. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[27] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate LiDAR semantic segmentation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4213–4220. IEEE, 2019.

[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[29] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[30] Nannan Qin, Xiangyun Hu, and Hengming Dai. Deep fusion of multi-view and multimodal representation of ALS pint cloud for 3D terrain scene recognition. *ISPRS journal of photogrammetry and remote sensing*, 143:205–212, 2018.

[31] Thiago Rateke, Karla Aparecida Justen, and Aldo von Wangenheim. Road surface classification with images captured from low-cost camera-road traversing knowledge (RTK) dataset. *Revista de Informática Teórica e Aplicada*, 26(3):50–64, 2019.

[32] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3D representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017.

[33] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[34] Laurent Smadja, Jérôme Ninot, Thomas Gavrilovic, et al. Road extraction and environment interpretation from LiDAR sensors. *IAPRS*, 38(281-286):1, 2010.

[35] M Soilán, A Nóvoa, A Sánchez-Rodríguez, B Riveiro, and P Arias. Semantic segmentation of point clouds with Pointnet and Kpconv architectures applied to railway tunnels. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2:281–288, 2020.

[36] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. SPLATNet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2530–2539, 2018.

[37] Weikai Tan, Nannan Qin, Lingfei Ma, Ying Li, Jing Du, Guorong Cai, Ke Yang, and Jonathan Li. Toronto-3D: A large-scale mobile LiDAR dataset for semantic segmentation of urban roadways. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 202–203, 2020.

[38] Lyne Tchapmi, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3D point clouds. In *2017 international conference on 3D vision (3DV)*, pages 537–547. IEEE, 2017.

[39] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[40] Anh-Vu Vo, Linh Truong-Hong, Debra F Laefer, and Michela Bertolotto. Octree-based region growing for point cloud segmentation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 104:88–100, 2015.

[41] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10296–10305, 2019.

[42] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph CNN for learning on point clouds. *ACM Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[43] Elyta Widyaningrum and Roderik C Lindenbergh. Skeleton-based automatic road network extraction from an orthophoto colored point cloud. *The 40th Asian onference on Remote Sensing (ACRS 2019), Daejeon, Korea*, 2019.

[44] Elyta Widyaningrum, Qian Bai, Marda K Fajari, and Roderik C Lindenbergh. Airborne laser scanning point cloud classification using the DGCNN deep learning method. *Remote Sensing*, 13(5):859, 2021.

[45] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893. IEEE, 2018.

[46] Yihuan Zhang, Jun Wang, Xiaonian Wang, and John M Dolan. Road-segmentation-based curb detection method for self-driving via a 3D-LiDAR sensor. *IEEE transactions on intelligent transportation systems*, 19(12):3981–3991, 2018.

[47] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537, 2015.

[48] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.

A

Extra Experiments

We compare the performance of three point-wise neural networks, i.e., PointCNN[1], GACNet[2], and RandLA-Net, on road type classification of MLS point clouds in **Study Area 1**. All three networks follow an Encoder-Decode structure and aggregate local information of each point within its K-Nearest Neighbors considering different importance of neighboring points. Different from PointCNN and GACNet, RandLA-Net constructs an enhanced geometric feature vector using both original coordinates and relative coordinates in the point neighborhood. Also, compared to PointCNN and GACNet, which adopts farthest sampling in encoding layers, RandLA-Net uses random sampling to ensure the processing efficiency.

During the training of PointCNN and GACNet, the input point cloud data is firstly split into regularly aligned blocks, with a block size of 50 m, to form smaller data batches. Each data batch contains 16384 points. By contrast, a training sample fed into RandLA-Net has totally 65535 points and is cropped from the input point clouds on the fly, with the central point randomly sampled. Moreover, for all networks, we use $K = 16$ nearest neighbors of each point and features (x, y, z, R, G, B) as input.

We present the quantitative results of classifying road point clouds into different usage types using PointCNN, GACNet, and RandLA-Net in Table A.1 and A.2. Note that we keep the original 9 classes of FEAT_TYPE in these experiments. Apparently, RandLA-Net achieves the best overall classification performance on **Study Area 1**, with an mIoU approximxately 10% higher than PointCNN and 20% higher than GACNet. From the per-class IoU results shown in Table A.2 we observe the dominant advantage of using RandLA-Net to classify *cycling path*, *island without traffic*, and *other*. For *sidewalk* and *motorway*, which have the largest number of points in our dataset, PointCNN has the best performance and obtains around 1% higher IoU than RandLA-Net. Compared to PointCNN and RandLA-Net, GACNet has the lowest quantitative results, especially for *rail track*. Notably, since we use a different deep learning framework (i.e., TensorFlow[3]) for GACNet, the performance might be affected.

	mA	mIoU
PointCNN	52.4%	44.7%
GACNet	41.6%	33.2%
RandLA-Net	**68.3%**	**53.1%**

Table A.1: Mean per-class accuracy (mA) and mean Intersection over Union (mIoU) results of road type (**FEAT_TYPE**) classification using PointCNN, GACNet, and RandLA-Net on **Study Area 1**, with (x, y, z, R, G, B) as input. FEAT_TYPE contains 9 road types.

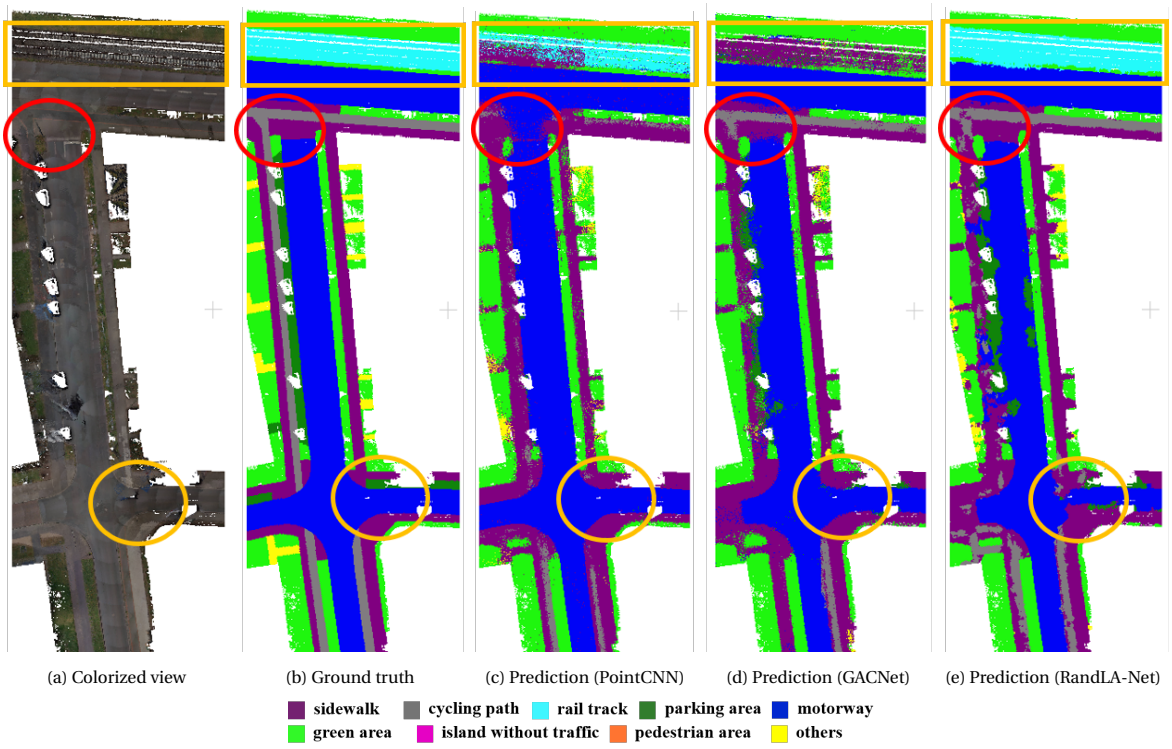
Figure A.1 further illustrates the differences among road type classification using PointCNN, GACNet, and RandLA-Net. The boxed area shows that GACNet fails to identify *rail track*. RandLA-Net detects *rail track* points well, but causes confusion near the boundary between *rail track* and *motorway*. The orange circle in

[1] PyTorch implementation from ArcGIS API for Python is used. Details can be found in https://developers.arcgis.com/python/api-reference/arcgis.learn.toc.html#arcgis.learn.PointCNN.
[2] Official TensorFlow implementation from https://github.com/wleigithub/GACNet is used.
[3] https://www.tensorflow.org/

	sidewalk	cycling path	rail track	parking area	motorway	green area	island	pedestrian area	other
PointCNN	**56.5%**	28.3%	70.6%	21.3%	**83.3%**	81.5%	41.5%	0	19.5%
GACNet	49.3%	31.4%	0	26.7%	80.9%	77.4%	23.0%	0	10.3%
RandLA-Net	55.1%	**50.0%**	**75.3%**	**33.5%**	82.2%	**83.3%**	**62.0%**	**7.1%**	**29.4%**

Table A.2: IoU of each **FEAT_TYPE** class using PointCNN, GACNet, and RandLA-Net on **Study Area 1**, with (x, y, z, R, G, B) as input. Island: *island without traffic*

Figure A.1e also highlights the confusion between *sidewalk* and *motorway* when using RandLA-Net. Compared to RandLA-Net, PointCNN achieves better classification for *motorway* points, but has problems in classifying *parking area* and narrow *cycling path* (see the red circled area in Figure A.1c). Considering the overall performance as well as results of each class, we finally select RandLA-Net as the backbone for road type classification of MLS point clouds in this research.



(a) Colorized view (b) Ground truth (c) Prediction (PointCNN) (d) Prediction (GACNet) (e) Prediction (RandLA-Net)

Figure A.1: Road type classification for **FEAT_TYPE** on **Study Area 1** using PointCNN, GACNet, and RandLA-Net.

B

ISPRS Paper

The following paper will be presented during the 2021 Digital Event of the XXIVth ISPRS Congress from 5 July to 9 July 2021.

ROAD TYPE CLASSIFICATION OF MLS POINT CLOUDS USING DEEP LEARNING

Q. Bai[1, 2*], R. C. Lindenbergh[1], J. Vijverberg[2], J. A. P. Guelen[2]

[1] Dept. of Geoscience and Remote Sensing, Delft University of Technology, The Netherlands
- (, r.c.lindenbergh@tudelft.nl)
[2] Cyclomedia Technology - (QBai, JVijverberg, JGuelen)@cyclomedia.com

Commission II, WG II/3

KEY WORDS: Mobile mapping, point clouds, semantic segmentation, road type, local feature aggregation, deep learning.

ABSTRACT:

Functional classification of the road is important to the construction of sustainable transport systems and proper design of facilities. Mobile laser scanning (MLS) point clouds provide accurate and dense 3D measurements of road scenes, while their massive data volume and lack of structure also bring difficulties in processing. 3D point cloud understanding through deep neural networks achieves breakthroughs since PointNet and arouses wide attention in recent years. In this paper, we study the automatic road type classification of MLS point clouds by employing a point-wise neural network, RandLA-Net, which is designed for consuming large-scale point clouds. An effective local feature aggregation (LFA) module in RandLA-Net preserves the local geometry in point clouds by formulating an enhanced geometric feature vector and learning different point weights in a local neighborhood. Based on this method, we also investigate possible feature combinations to calculate neighboring weights. We train on a colorized point cloud from the city of Hannover, Germany, and classify road points into 7 classes that reveal detailed functions, i.e., *sidewalk*, *cycling path*, *rail track*, *parking area*, *motorway*, *green area*, and *island without traffic*. Also, three feature combinations inside the LFA module are examined, including the geometric feature vector only, the geometric feature vector combined with additional features (e.g., color), and the geometric feature vector combined with local differences of additional features. We achieve the best overall accuracy (86.23%) and mean IoU (69.41%) by adopting the second and third combinations respectively, with additional features including Red, Green, Blue, and intensity. The evaluation results demonstrate the effectiveness of our method, but we also observe that different road types benefit the most from different feature settings.

1. INTRODUCTION

Automation of road information extraction is of great significance to economic and social development. Road type, which indicates the function of a road segment, is key to various applications, including autonomous driving, inspections of infrastructures, and decision making of companies and governments (Zhu et al., 2012). LiDAR provides accurate 3D point measurements and is illumination invariant, showing a strong ability for mapping. Similar to image recognition, point cloud processing also benefits from the rapid development of deep learning techniques (Liu et al., 2019). However, it is still challenging to interpret 3D point clouds using neural networks due to their irregular data structure.

Determining the type of each road point is consistent with the aim of point cloud semantic segmentation. Recent studies on semantic segmentation of point clouds using deep learning mainly consist of two kinds of methods, i.e., projection-based and point-based methods. In projection-based methods, point clouds are first projected onto 2D planes (i.e., images) (Wu et al., 2018) or converted into voxels (i.e., 3D grids) (Riegler et al., 2017). Through achieving a regularly aligned data format, 2D or 3D convolutional neural networks (CNN) can be applied. Although these methods address the problem of unorganized point clouds indirectly, some spatial information is lost and additional computational resources are needed during pre-processing.

As an active research topic in this area, point-based neural networks can directly consume and model 3D point data. PointNet (Qi et al., 2017a), the pioneering work among these methods, employs a series of shared multi-layer perceptrons (MLP) to learn higher-dimensional features for each point. Then these per-point features are aggregated by applying a symmetric function (e.g., max-pooling), ensuring that point cloud processing is irrelevant to the point order. However, PointNet does not consider local structures inside the point cloud, limiting its performance in complex scenes (Qi et al., 2017b). Starting from PointNet, many networks are proposed combining MLPs with local feature aggregation. The local feature aggregation module aims to extract prominent features from a point neighborhood, thereby exploiting wider contextual information around each point. The choice of input features for local aggregation has a great impact on its effectiveness. PointNet++ uses relative coordinates in a local region together with additional point features (e.g., R, G, B), while DGCNN (Wang et al., 2019) constructs the concatenation of all original and relative features as input. RandLA-Net (Hu et al., 2020), which is adopted in this study, employs more complex encoding for relative coordinates to capture geometric details. The encoded geometric feature, together with additional point features, is then used to achieve local feature aggregation. Different from PointNet++ and DGCNN, which use a symmetric function to aggregate input features indistinguishably, RandLA-Net represents local information as a weighted sum of all neighboring point features, making the choice of input features even more crucial for capturing the local geometry.

Some of the aforementioned methods have already verified their model performance on benchmark MLS point cloud datasets

* Corresponding author

like SemanticKITTI (Behley et al., 2019). The labeling of these datasets covers the whole road scene including road surface, cars, buildings, etc. However, further research on detailed classification focusing on different road types is still needed. Some studies also evaluate different input features of the local aggregation module (Widyaningrum et al., 2021). It turns out that one fixed feature combination is not optimal for all datasets (Liu et al., 2020). To find a proper setting for road type classification, it is important to conduct more experiments.

The main contributions of this paper are:

- We achieve detailed road type classification in dense urban areas by applying RandLA-Net.
- We assess how features should be combined to achieve weights of neighboring points when aggregating local information in point clouds.

This paper is organized as follows: Section 2 illustrates the dataset employed in this study. Section 3 describe the methodology, including the data pre-processing procedure, details of the neural network, and adopted evaluation metrics. Experiment results are discussed in Section 4. Finally, Section 5 presents the drawn conclusions.

2. DATASET AND STUDY AREA

The MLS point cloud used in this paper is acquired by Cyclomedia's proprietary recording system (Cyclomedia, 2021), in the city of Hannover, Germany. This system is mainly composed of 5 high-resolution cameras and a Velodyne HDL-32E LiDAR sensor (Velodyne, 2010). Figure 1 shows the trajectory of the recording vehicle, which is about 16 km in length. The original LiDAR point cloud has an average point spacing of 1 cm and is colorized by panoramic images obtained at the same time.

Ground truth annotations of the MLS point cloud contain 9 road classes: *sidewalk*, *cycling path*, *rail track*, *parking area*, *motorway*, *green area*, *island without traffic*, *pedestrian area* (car-free zones) and *others*. The order of these classes also reveals the priority of labeling. For example, if a motorway is crossing a rail track in a point cloud, corresponding points will be labeled as *rail track*.

3. METHODOLOGY

This study investigates the capability of a deep neural network, i.e., RandLA-Net, for classifying 3D point clouds into different road types and evaluates the performance of different feature combinations in the local feature aggregation module. Our methodology mainly includes data pre-processing, training with RandLA-Net, and evaluation.

3.1 Data pre-processing

To handle the sheer volume of the acquired MLS point cloud, we first downsample it using grid sampling, with a grid size of 0.1 m. Figure 2a presents an example of the colorized point cloud after downsampling. Afterwards, non-road points are removed by a ground filtering approach (Isenburg, 2014), as shown in Figure 2b. The label of each point is then achieved through overlaying the ground truth annotations, which are polygons stored in the shapefile format, on the road point cloud

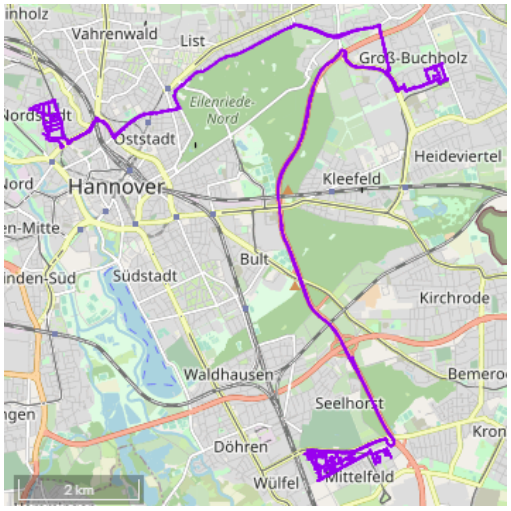


Figure 1. Trajectory of the recording vehicle shown in purple, with a length about 16 km. The base map is provided by © OpenStreetMap.

(see Figure 2d). Also, points belonging to *pedestrian area* and *others* have similar appearance as *sidewalk*. Considering that they are detected with the help of other information like a road sign in practice, *pedestrian area* and *others* are merged into the *sidewalk* to ease the training.[(1)]

After pre-processing, the point cloud dataset used for our study has a total number of 74,629,166 points, with 9 attributes, i.e., x, y, z, R, G, B, intensity, return number, and number of returns. Besides, the distribution of points in 7 road types is illustrated in Table 1, in which a class imbalance issue can be observed. *Motorway* contains a dominant number of points. *sidewalk* and *green area* are also frequently seen in the data. By contrast, *rail track* has the least amount of points. The MLS point cloud after pre-processing is vertically split into 39 tiles, with 29 tiles for training and 10 for testing.

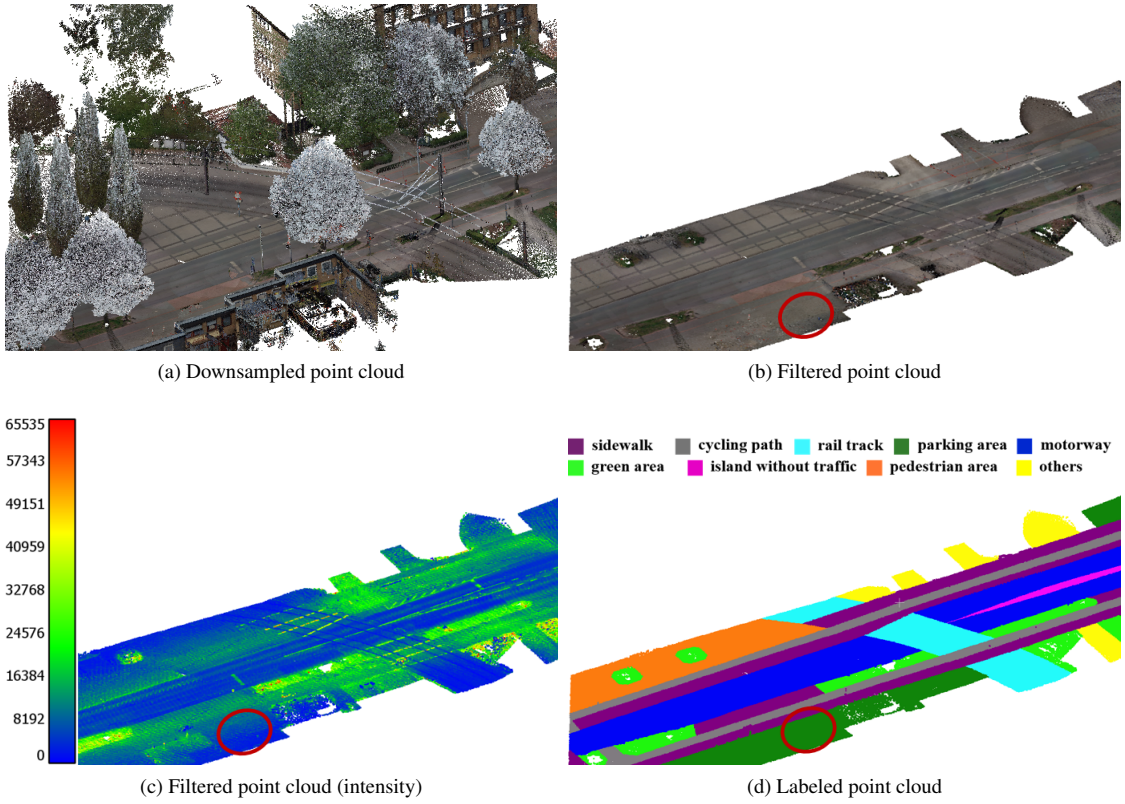
sidewalk	cycling path	rail track	parking area
17.27	3.22	0.54	3.92
motorway	green area	island without traffic	
32.68	14.12	2.88	

Table 1. Number of points ($\cdot 10^6$) in each road type.

3.2 RandLA-Net

We implement RandLA-Net (Hu et al., 2020) for road type classification in this study. RandLA-Net is a point-wise neural network and follows an encoder-decoder hierarchical design (see Figure 3). Given a point cloud with a large number of points, the points are progressively downsampled in each encoding layer and upsampled again in decoding layers to preserve the original resolution in final predictions. To achieve processing efficiency, random sampling is chosen as the downsampling strategy. Since random sampling drops points non-selectively, each neural layer also contains an effective local feature aggregation (LFA) module to summarize neighborhood information without losing important point features. The LFA module

[(1)] When using (x, y, z, R, G, B) and the original feature combination in the LFA module of RandLA-Net, the mean IoU is improved by 10.97% after merging the labels.



(a) Downsampled point cloud (b) Filtered point cloud

(c) Filtered point cloud (intensity) (d) Labeled point cloud

Figure 2. Downsampled, filtered and labeled point cloud. Circle areas highlight the similarity between some parking areas and sidewalks.

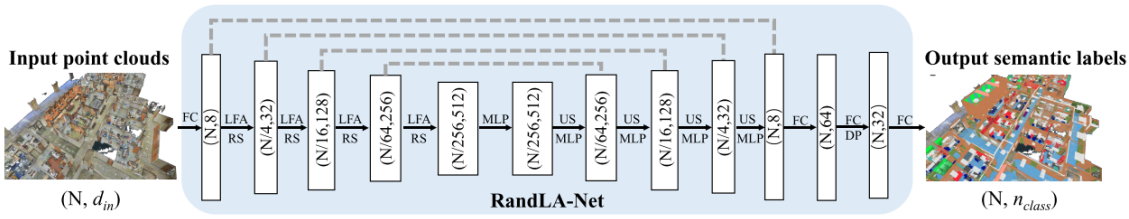


Figure 3. Overview of the RandLA-Net architecture (Hu et al., 2020). N indicates the number of input points. FC: Fully Connected layer, LFA: Local Feature Aggregation, RS: Random Sampling, MLP: shared Multi-Layer Perceptron, US: Up-sampling, DP: Dropout.

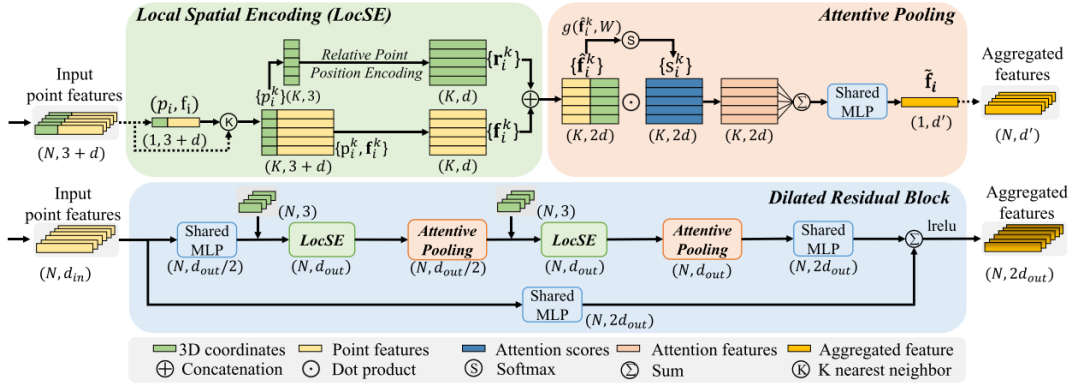


Figure 4. Components of an encoding layer in RandLA-Net (Hu et al., 2020). Top: Local Spatial Encoding (LocSE) block which transforms the input features and Attentive Pooling block which aggregates the local information based on weighing the neighboring points. Bottom: Two pairs of LoSE and Attentive Pooling blocks are stacked together to increase the receptive field, which forms the Dilated Residual Block of each encoding layer.

is the key to modeling and perceiving the local geometry of point clouds. Moreover, the neighborhood around each point is selected using K-Nearest Neighbor (KNN) in RandLA-Net.

As shown in Figure 4 (top), the LFA module consists of two components, i.e., Local Spatial Encoding (LocSE) and Attentive Pooling. Within LocSE, coordinates of the input points are first transformed to a higher dimensional geometric feature vector r_i^k according to:

$$r_i^k = MLP(p_i \oplus p_i^k \oplus (p_i - p_i^k) \oplus \|p_i - p_i^k\|), \quad (1)$$

where MLP = multi-layer perceptrons
$i \in \{1, 2, ..., N\}$
N = the total number of points
$k \in \{1, 2, ..., K\}$
K = the number of nearest neighbors
p_i = coordinates of the centered point
p_i^k = coordinates of one neighboring point
$\oplus$ = concatenation operation
$\|\cdot\|$ = Euclidean distance

The geometric feature vector r_i^k and additional features f_i^k (e.g, R, G, B) are then concatenated as $\hat{f}_i^k$, which is the input of Attentive Pooling.

The aim of Attentive Pooling is to aggregate the enhanced point feature $\hat{f}_i^k$ in the neighborhood to achieve local contextual information for each point. Neural networks like PointNet++ and DGCNN apply a symmetric function (e.g., max-pooling and $\sum$) as the aggregation function, which is simple but inevitably processes the neighboring points indistinguishably, causing a certain loss of geometric information. The Attentive Pooling in RandLA-Net, instead, learns different weights s_i^k of the neighboring points through a MLP, as indicated by $g(\hat{f}_i^k, W)$ in Figure 4 (top). The neighborhood features are subsequently aggregated by taking a weighted sum. Moreover, RandLA-Net applies the LFA module twice in each layer to effectively increase the receptive field of the network, as shown in Figure 4 (bottom).

In this study, we use a colorized MLS point cloud. Apart from the geometric vector r_i^k, how to combine additional features (e.g., R, G, B) to aggregate local information in road scenes remains to be discussed. In urban areas, road objects like *motorway* and *green area* have totally different appearance. Their variation in color can also differ a lot. Additionally, as indicated in Figure 2c, intensity values of the vegetation (*green area*) present distinct characteristics. Thus, it might be beneficial to include these features or even their local differences as additional information sources to help distinguish road types.

However, it may happen that the surface material of two adjacent road segments (e.g., *sidewalk* and *parking area* shown in circled areas of Figure 2) are the same, making the appearance and reflection values of different road objects very similar. In this case, it is possible that using only geometric features can reduce class confusion and acquire more accurate results.

Based on these assumptions, we compare three feature combinations to calculate neighboring weights in the local feature aggregation module of RandLA-Net, which refer to the choice of $\hat{f}_i^k$ in $g(\hat{f}_i^k, W)$:

1. $\mathbf{r_i^k}$: Geometric feature vector only.
2. $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$: Geometric feature vector r_i^k concatenated with additional features f_i^k, which is the original implementation of RandLA-Net.
3. $\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$: Geometric feature vector r_i^k concatenated with relative additional features $(f_i - f_i^k)$.

We also consider two settings of the additional features f_i^k, i.e., (R, G, B) and (R, G, B, I), with I indicating the intensity. The intensity feature is a more stable attribute compared to RGB values since it is not affected by illumination conditions during recording.

3.3 Evaluation metrics

To evaluate and compare the performance of different feature combinations illustrated in Section 3.2, we determine the following evaluation metrics in this study, which are commonly used in the semantic segmentation task:

- Overall accuracy (OA), which measures the proportion of correctly classified points among all input points.
- Mean Intersection over Union (mIoU), which is the mean value of Intersection over Union (IoU) in each class, with IoU defined as:

$$\text{IoU} = \frac{\text{Overlap of the predicted and ground truth}}{\text{Union of the predicted and ground truth}}. \quad (2)$$

4. RESULTS AND DISCUSSION

In this section, we first compare the evaluation results for three feature combinations in the local feature aggregation module of RandLA-Net in Section 4.1. Section 4.2 shows the impact of adding intensity features on the overall performance, as well as the results of several specific road types. Finally, we discuss the importance of defining appropriate road classes that represent distinct functions in Section 4.3.

Table 2 and Table 3 summarize the quantitative results of road type classification with different feature combinations in the LFA module.

	$\mathbf{f_i^k}$	OA	mIoU
$\mathbf{r_i^k}$	RGB	84.01%	67.05%
$\mathbf{r_i^k} \oplus \mathbf{f_i^k}$		83.58%	64.07%
$\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$		**85.66%**	**69.17%**
$\mathbf{r_i^k}$	RGBI	85.57%	68.75%
$\mathbf{r_i^k} \oplus \mathbf{f_i^k}$		**86.23%**	68.26%
$\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$		86.09%	**69.41%**

Table 2. Comparison of the overall accuracy (OA) and the mean Intersection over Union (mIoU) among different setups in the LFA module.

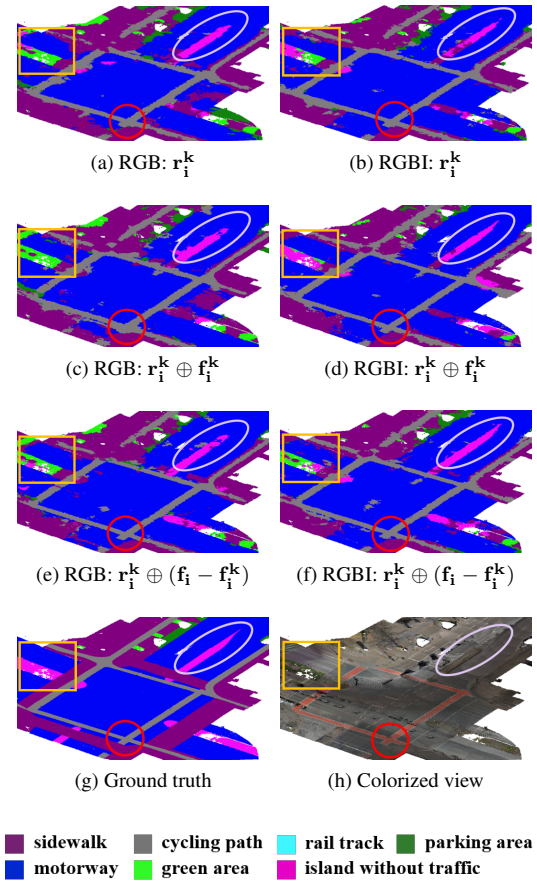
Figure 5 and Figure 6 also illustrate part of the results in our test area. Compared to the ground truth labeling, each feature setup achieves reasonable predictions in general. However, since RandLA-Net learns and aggregates point features in a local neighborhood, inaccurate geometric shapes along object edges can be observed. Moreover, a feature combination that improves the overall accuracy does not ensure performance gains on each road type.

	$\mathbf{f_i^k}$	sidewalk	cycling path	rail track	parking area	motorway	green area	island without traffic
$\mathbf{r_i^k}$	RGB	64.7%	50.6%	**87.1%**	31.2%	82.2%	85.4%	68.2%
$\mathbf{r_i^k} \oplus \mathbf{f_i^k}$		63.4%	48.6%	75.7%	32.8%	82.3%	83.5%	62.3%
$\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$		**67.5%**	**57.9%**	85.6%	**35.2%**	**83.6%**	**85.5%**	**68.9%**
$\mathbf{r_i^k}$	RGBI	67.3%	50.4%	82.0%	34.2%	85.1%	**87.0%**	**75.3%**
$\mathbf{r_i^k} \oplus \mathbf{f_i^k}$		**68.3%**	**53.5%**	82.7%	**36.3%**	**86.0%**	85.5%	65.5%
$\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$		67.6%	53.2%	**87.0%**	34.6%	85.4%	86.9%	71.1%

Table 3. IoU of each class among different setups in the local feature aggregation module.

4.1 Comparison between different feature combinations

In the case of using RGB features, neighboring weights obtained with the combination of geometric feature vector r_i^k and local feature differences $(f_i - f_i^k)$ result in a dominant advantage in both evaluation metrics. Adding intensity, $\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$ helps to achieve the best mIoU of 69.41%, but the gap between it and other feature combinations is much smaller than that shown when only adopting RGB features.



(a) RGB: $\mathbf{r_i^k}$ (b) RGBI: $\mathbf{r_i^k}$

(c) RGB: $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$ (d) RGBI: $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$

(e) RGB: $\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$ (f) RGBI: $\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$

(g) Ground truth (h) Colorized view

Figure 5. Comparison of road type classification results using different feature combinations to weigh the neighboring points. Holes in the dataset are caused by the removal of cars in pre-processing. Rectangles, ellipses, and circles highlight the differences between results with different feature combinations and the ground truth.

As illustrated in Table 3, the best performance on *cycling path* is achieved when combining geometric feature vector and color difference. The red circled area in Figure 5h shows part of a cycling path painted in two colors. RGB difference in the local region helps to highlight the color variation within one object and produces a clear outline of the cycling path in Figure 5e. However, both the cycling path and motorway in this figure are made of asphalt, so involving the local difference of intensity (i.e., $\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$) does not bring an advantage compared to using original intensity values (i.e., $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$), which is also supported by the IoU results in Table 3.

Moreover, Table 3 demonstrates the effectiveness of using the geometric feature r_i^k only in the classification of *green area* and *island without traffic*. As shown in the boxed area of Figure 6h, there is a vegetation stripe next to the southern border of the rail track. Figure 6b indicates difficulties in distinguishing both classes. Due to the illumination condition, the hue of this figure is slightly dark, reducing the contrast in the appearance of *green area* and *rail track*. Eliminating the effect of RGB features when weighing neighboring points helps to highlight the difference in geometrical shapes of objects (see Figure 6a).

For the class *island without traffic*, using only the geometric vector r_i^k shows a dominant advantage. *Island without traffic* refers to areas that channel traffic, which is always slightly higher than the surrounding road surface. As shown in the white circled area in Figure 5, the traffic island has a very similar color as the motorway, which brings confusion in Figure 5c.

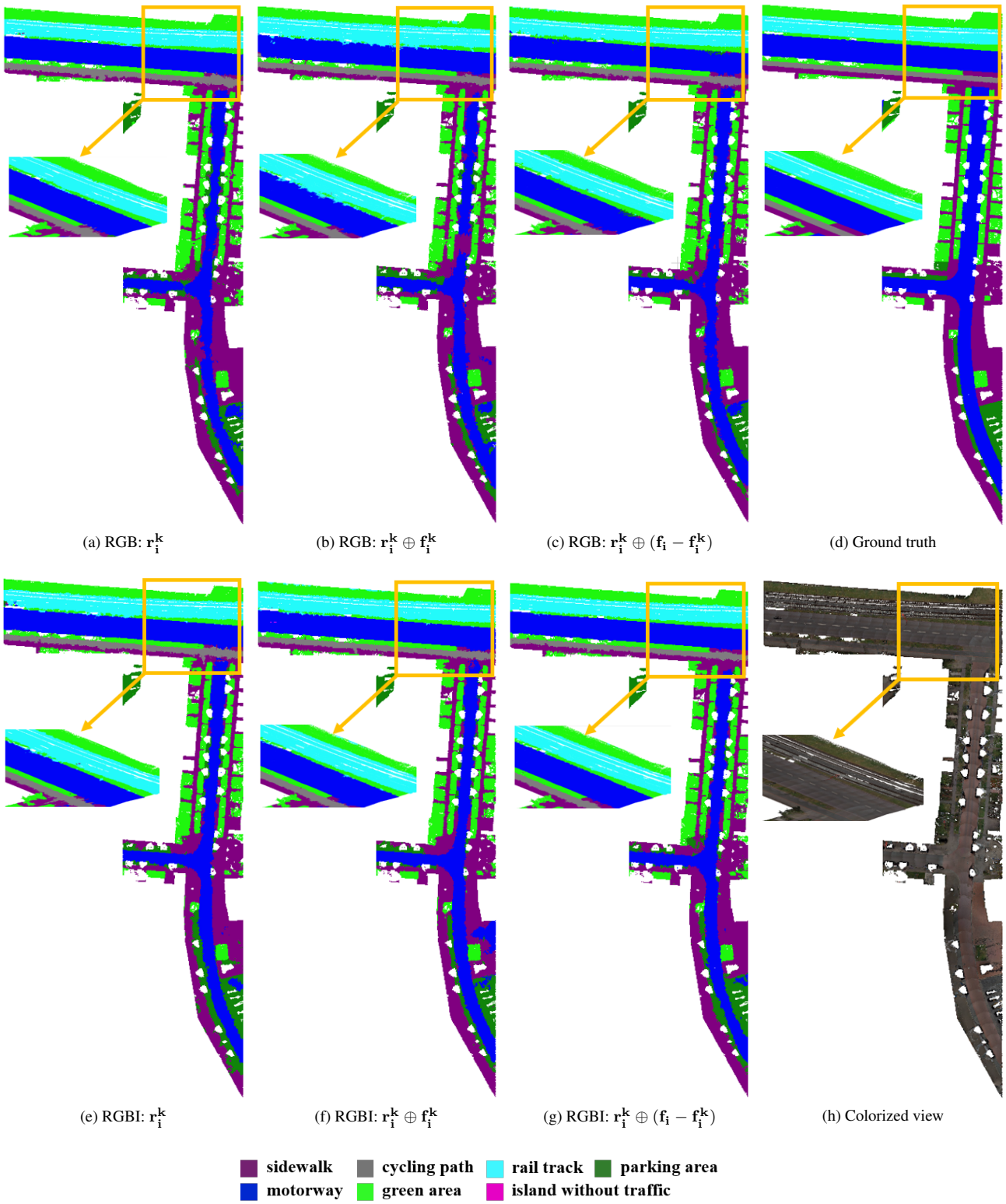
Also, one can see that only the geometric feature vector does not provide enough information for the network when adjacent objects are made of the same material but have a difference in color, especially for classes (e.g., *parking area*) that are sometimes identified by paintings in specific colors.

However, the segmentation performance on the *motorway* class is only slightly affected by the feature combination of weighing the neighboring points, which can also be explained by the object properties. *Motorway* has the most simple geometric characteristics among all these classes and is more invariant than additional features like RGB.

4.2 Impact of intensity

Comparisons of mIoU in Table 2 suggest that adding intensity features is beneficial when classifying different road types of 3D point clouds. Intensity brings effective information in training the model. Only relying on RGB features is not enough to distinguish some classes. First, there exist some traffic islands that are covered with vegetation (see boxed areas in Figure 5), resulting in *island without traffic* misclassified as *green area* if only RGB features are used to weigh neighboring points.

Also, point colors are easily affected by the change of illumination (see Figure 7a), while intensity values are more stable in case of shadows (see Figure 7b). Classification results in Figure 7c indicates that shadows cause confusions between the *sidewalk* and *cycling path* with additional features (R, G, B). Such confusions are largely reduced in Figure 7d, when the intensity feature is also considered.

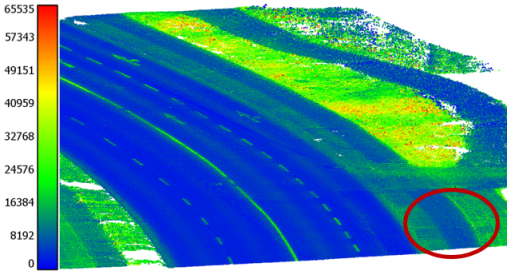


(a) RGB: $\mathbf{r_i^k}$ (b) RGB: $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$ (c) RGB: $\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$ (d) Ground truth

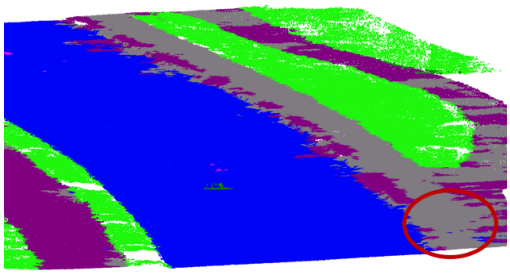
(e) RGBI: $\mathbf{r_i^k}$ (f) RGBI: $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$ (g) RGBI: $\mathbf{r_i^k} \oplus (\mathbf{f_i} - \mathbf{f_i^k})$ (h) Colorized view

Figure 6. Comparison of road type classification results using different feature combinations to weigh the neighboring points.

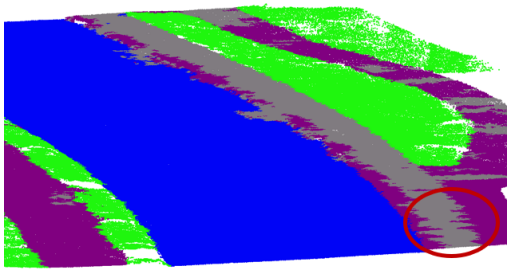
(a) Colorized view



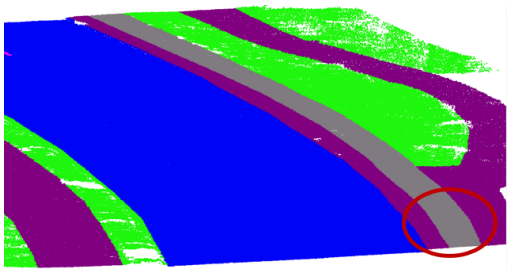
(b) Intensity



(c) RGB: $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$



(d) RGBI: $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$



(e) Ground truth

Figure 7. Comparison of road type classification results with additional features (R, G, B) and (R, G, B, I). Circle areas indicate the impact of the intensity feature.

In the case of *sidewalk*, *parking area*, and *motorway*, the original feature combination (i.e., $\mathbf{r_i^k} \oplus \mathbf{f_i^k}$) in the LFA module of RandLA-Net gives the best result when intensity is also used as input for the network, as indicated in Table 3. This tells us that intensity has a larger impact on the performance of these classes than the choice of feature combination in local information aggregation.

4.3 Definition of road types

As discussed in previous sections, some road classes in our dataset have the same material type or even appearance, which confuses the classification task to some extent. For instance, the vertical motorway in Figure 6 looks very similar to the sidewalk next to it. The horizontal motorway in this figure, on the other hand, has a different color. Moreover, in our dataset there exists a priority list in labeling, e.g., a road object should be classified as *sidewalk* even though it is also used as *motorway* (see 5), which is due to the importance of promoting green transportation in large cities nowadays.

Indeed, when defining the road type, the usage of a road segment is the most meaningful for the human being and practical applications like urban planning. However, a road class definition with high complexity might harm the performance of deep neural networks.

5. CONCLUSIONS

In this study, a deep neural network designed for the semantic segmentation of large-scale point clouds, RandLA-Net, is employed to classify road types of a colorized MLS point cloud. Considering the key component in RandLA-Net, which is the local feature aggregation (LFA) module, three feature combinations used to calculate point weights in a local neighborhood are assessed and compared. The difference in using RGB and RGBI features in road type classification is also discussed.

Through our experiments, RandLA-Net is demonstrated to be applicable to the road type classification task. The best mIoU (69.41%) is achieved when combining the enhanced geometric feature vector and local differences of RGBI features. The geometric feature vector adopted by RandLA-Net is powerful in modeling the 3D geometry and learning the local shapes of road objects, especially *island without traffic*. Using feature difference instead of the feature itself (which is the original implementation of RandLA-Net) makes it easier to detect complex objects in our dataset, like *cycling path* painted in various colors. Moreover, intensity, an important LiDAR feature, adds effective information to the neural network and helps to overcome the negative effect of illumination changes in the environment, which improves the overall performance of RandLA-Net.

In the pre-processing step, we apply grid sampling with a grid size of 0.1 m, which helps to avoid the problem of varying densities in point clouds and does not harm the local structure of road segments. As future work, more investigation on the effect of downsampling strategies can be conducted. Also, although RandLA-Net aims to process neighboring points indistinguishably through learning different weights, there is still space in improving the delineation between objects in the classification results. The feasibility of RandLA-Net on larger datasets and comparisons to other methods (e.g., image-based methods) should also be further studied. Additionally, urban scenes designed for modern life always show complex characteristics,

bringing difficulties to the automatic detection of objects like road segments. Definition of the road types determines information input to deep neural networks and affects how the scene is modeled. Dividing the road classes in a balanced way, to account for both the test accuracy and practical usage, needs a more detailed discussion in future research.

REFERENCES

Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J., 2019. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*.

Cyclomedia, 2021. Capture data from public spaces. https://www.cyclomedia.com/us/product/data-capture/data-capture. (Accessed on April 2021).

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A., 2020. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Isenburg, M., 2014. LAStools - Efficient tools for LiDAR processing. http://rapidlasso.com/LAStools. (Accessed on February 2021).

Liu, W., Sun, J., Li, W., Hu, T., Wang, P., 2019. Deep learning on point clouds and its application: A survey. *Sensors*, 19(19), 4188.

Liu, Z., Hu, H., Cao, Y., Zhang, Z., Tong, X., 2020. A closer look at local aggregation operators in point cloud analysis. *European Conference on Computer Vision*, Springer, 326–342.

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 77–85.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. NIPS'17, Curran Associates Inc.

Riegler, G., Osman Ulusoy, A., Geiger, A., 2017. Octnet: Learning Deep 3D Representations at High Resolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3577–3586.

Velodyne, 2010. HDL-32E: High Resolution Real-Time 3D Lidar Sensor. https://velodynelidar.com/products/hdl-32e/. (Accessed on April 2021).

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., Solomon, J. M., 2019. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transactions On Graphics (tog)*, 38(5), 1–12.

Widyaningrum, E., Bai, Q., Fajari, M. K., Lindenbergh, R. C., 2021. Airborne Laser Scanning Point Cloud Classification Using the DGCNN Deep Learning Method. *Remote Sensing*, 13(5). https://www.mdpi.com/2072-4292/13/5/859.

Wu, B., Wan, A., Yue, X., Keutzer, K., 2018. Squeezeseg: Convolutional Neural Nets with Recurrent CRF for Real-time Road-object Segmentation from 3D LiDAR Point Cloud. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 1887–1893.

Zhu, Q., Chen, L., Li, Q., Li, M., Nüchter, A., Wang, J., 2012. 3D LIDAR point cloud based intersection recognition for autonomous driving. *2012 IEEE Intelligent Vehicles Symposium*, 456–461.