DEEP LEARNING FOR SEMANTIC SEGMENTATION OF AIRBORNE LASER SCANNING POINT CLOUDS

Yaping Lin

DEEP LEARNING FOR SEMANTIC SEGMENTATION OF AIRBORNE LASER SCANNING POINT CLOUDS

DISSERTATION

to obtain
the degree of doctor at the University of Twente,
on the authority of the rector magnificus,
prof.dr.ir. A. Veldkamp,
on account of the decision of the Doctorate Board,
to be publicly defended
on Wednesday 7 September 2022 at 14.45 hours

by

Yaping Lin

born on the 12th of October, 1994
in Henan, China

This thesis has been approved by
Prof. dr.ir. M.G. Vosselman, supervisor
Dr. M.Y. Yang, co-supervisor

ITC dissertation number 417
ITC, P.O. Box 217, 7500 AE Enschede, The Netherlands

ISBN 978-90-365-5410-7
DOI 10.3990/1.9789036554107

Cover designed by Yaping Lin and Job Duim
Printed by CTRL-P, Hengelo

UNIVERSITY OF TWENTE.

FACULTY OF GEO-INFORMATION SCIENCE AND EARTH OBSERVATION

Graduation committee:

Chairman/Secretary

Prof.dr. F.D. van der Meer	University of Twente (ITC)

Supervisor

Prof.dr.ir. M.G. Vosselman	University of Twente (ITC)

Co-supervisor

Dr. M.Y. Yang	University of Twente (ITC)

Committee Members

Dr. F.C. Nex	University of Twente (ITC)
Prof.dr.ing. B. Rosic	University of Twente (AMDA)
Prof.dr. G. Xia	Wuhan University
Prof.dr. J.D. Wegner	University of Zurich

Acknowledgements

I did not expect that I would spend the next six years in Enschede when the first time I arrived at the Enschede train station with my friend Wenfei on the 10th of September, 2016. During two years of my master's study and four years of my Ph.D. study at ITC, I gradually fall in love with this small and quiet city which is an ideal place for doing research. Throughout this long journey, I received a lot of support from my supervisors, colleagues, friends and family and turned into a more experienced researcher.

First and foremost, I would like to express my deepest gratitude to my promotor and supervisor Prof. George Vosselman. Although George is quite busy, he is always willing to help even helping me with data pre-processing. He also helped me a lot with my paper writing to make them more readable and logical. He always takes a rigorous attitude towards my research and gives critical comments on my work to make it more solid. His sharp insights inspired me to think about my research from different aspects. I really appreciate his dedication to my Ph.D study.

I'm extremely grateful to my co-promotor Dr. Michael Ying Yang, who always encourages me to be ambitious with my research. Every time I meet difficulties in my research, he is there to help me find key issues and provide innovative ideas. Michael has broad research interests and is a visionary supervisor. He always encourages me to have an open mind to new ideas and never be afraid of trying new methods.

I would thank Dr. Francesco Nex for supervising my master thesis and Dr. Sander Oude Elberink and Dr. Claudio Persello for offering me a Postdoc position at ITC. I would also express my thanks to my ITC colleagues. Our secretaries Teresa and Jenny are always ready to help and are good at organizing EOS activities. Jenny also helped me a lot with proofreading. Thank Sofia for helping with the Dutch summary of my thesis. I also really enjoyed the chats with Ling Chang, Peng Jia, Parya, Wan, Ville, Barsha, Frank, Caroline, Andrea, Sophie, Samer, Diogo, Phillip, Shayan, Zill, Anurag, Abhisek and Yu-Lun.

Many thanks to my friends in the scene understanding group led by Michael. Zhenchao Zhang, Ye Lyu, Wentong Liao are always willing to share their experiences when I meet technical issues. In our group seminars, Yuren Cong, Yao Wei and Kun Li always ask interesting questions about my research and make me have further reflections on my work. I really appreciate those academic discussions but also the dinners and games after working hours with SUG friends.

I would like to show my sincere thanks to my Chinese friends at ITC. They are Yanwen Wang, Wufan Zhao, Fashuai Li, Mengmeng Li, Qian Lu, Ning Zhang, Cai Wu, Shaoqing Dai, Wen Zhou, Ting Zhou, Qilei Huang, Han Yue, Siyang Chen, Bin Zhang, Yao Li, Xu Zhang, Zhishuang Yang, Dayan Guan, Guizhong Fu, Ying Ao, Shan Huang, Yiwen Wang, Ruodan Zhuang, Wenyi Chu, Ruoshao Zeng and Xiaoshuai Xie. I really enjoyed the time with you, having parties and sharing delicious food. Many thanks to Wenfei Dong, my first friend at ITC. You and Kuan Chai organized great trips and took good care of me during the trips. Now you are going to have little Kuan and I'm sure you will be great parents! Also, thanks to my friends, Qi Zhang, Shanbing Chen, Chu Chu, Shuang Song and Ying Sun for video calls during nights and weekends.

I'm also grateful to Dr. Clement Mallet and Dr. Loic Landrieu for offering me an internship at IGN Paris. Loic always has clear insights into different methods and makes constructive suggestions on my experiments. I also want to thank my Chinese friends at IGN, Yizi Chen, Yuhao Jiang, Lulin Zhang and Teng Wu, who helped me quickly settle down in Paris. Many thanks to my IGN colleagues, Damien, Romain, Mohamed, Helen, Emile, Yanis, Ekaterina, and Luc who are willing to speak English to me, have academic discussions with me and tell me where to visit and where to eat in Paris.

Lastly, I would like to express my sincere thanks to my parents who always unconditionally trust and support me. Also, special thanks to my boyfriend and best friend Shaoyu Wang, who understands all my laughter and tears and is always on my side. You are the sunshine in my gloomy days.

Table of Contents

Acknowledgements ii
List of figures vi
List of tables ix
Chapter 1 – Introduction 1
1.1 Background 2
1.2 Airborne laser scanning point clouds 3
1.3 Advantage of deep learning techniques 4
1.4 Research gap 5
1.5 Research objectives 5
1.6 Outline 7
Chapter 2 – Local and Global Encoder Network for Semantic Segmentation of Airborne Laser Scanning Point Clouds 8
Abstract 9
2.1 Introduction 10
2.2 Related work 12
2.2.1 Traditional methods 12
2.2.2 Deep learning methods 13
2.2.3 Attention models 16
2.3 Method 17
2.3.1 Hybrid convolution block 17
2.3.2 SegECC 21
2.3.3 Spatial-channel attention 23
2.3.4 Overall Network architecture 26
2.4 Experiments 28
2.4.1 Experiments on ISPRS benchmark dataset 28
2.4.2 Experiments on DFC2019 dataset 42
2.5 Conclusion 45
Chapter 3 – Active and Incremental Learning for Semantic Airborne Laser Scanning Point Cloud Segmentation 46
Abstract 47
3.1 Introduction 48
3.2 Related work 50
3.2.1 Deep learning approaches 50
3.2.2 Active learning 52
3.2.3 Incremental learning 54
3.3 Method 55
3.3.1 Active learning 56
3.3.2 Query functions 58
3.3.3 Semantic point cloud segmentation by neutral networks 61
3.3.4 Incremental learning 62
3.4 Experiments 62
3.4.1 Data description 63

3.4.2 Preprocessing 64
3.4.3 Network implementation 66
3.4.4 Accuracy assessment 67
3.4.5 Active learning setup 67
3.4.6 Incremental learning setup 70
3.4.7 Results 71
3.5 Conclusion 82
Chapter 4 – Weakly Supervised Semantic Segmentation of Airborne Laser Scanning Point Clouds 84
Abstract 85
4.1 Introduction 86
4.2 Related work 89
4.2.1 Deep learning on point clouds 89
4.2.2 Weakly supervised semantic segmentation on 2D images. 92
4.2.3 Deep learning on point clouds with fewer annotation efforts 92
4.3 Method 94
4.3.1 Brief review on MPRM 95
4.3.2 Pseudo label generation 98
4.3.3 Training with pseudo labels 102
4.4 Experiments 104
4.4.1 Experiments on ISPRS benchmark dataset 104
4.4.2 Experiments on Rotterdam dataset 122
4.4.3 Experiments on DFC dataset 127
4.5 Conclusion 132
Chapter 5 – Synthesis 133
5.1 Conclusions per Objective 134
5.2 Reflections and Outlook 136
Bibliography 140
Summary 151
Samenvatting 153
Author's Biography 156
ITC Dissertation List 157

List of figures

Figure 1.1 Principle of Lidar (Vosselman and Maas, 2010). 3
Figure 1.2 The structure of the research objectives in this thesis. 6
Figure 2.1 Illustration of kernel points distribution for 2D and 3D KPConv. 20
Figure 2.2 The convolutional block used in Thomas et al. (2019) (top) and the hybrid 2D-3D block used in this chapter (bottom). 20
Figure 2.3 Steps in SegECC to obtain segment embeddings using edged conditioned convolutions (ECC). 21
Figure 2.4 The structure of the hybrid-SegECC block. 22
Figure 2.5 Structures of spatial attention (top) and channel attention (bottom). 25
Figure 2.6 Structure of spatial-channel attention model. 25
Figure 2.7 Illustration of the proposed LGENet architecture for semantic segmentation of ALS point clouds. 27
Figure 2.8 An overview of the ISPRS benchmark dataset. 29
Figure 2.9 Classification results of our LGENet on the ISPRS benchmark dataset. 31
Figure 2.10 The error map of our LGENet on the ISPRS benchmark dataset. 32
Figure 2.11 Qualitative classification results on ISPRS benchmark dataset with different regularization factors. 38
Figure 2.12 Examples of segmentation results on ISPRS benchmark dataset. 39
Figure 2.13 Distribution of neighbourhood size for Vaihingen dataset. 40
Figure 2.14 Qualitative comparison of model performance with spatial-channel attention and without spatial-channel attention. 41
Figure 2.15 Some examples of classification results on the DFC2019 dataset obtained from different models. 43
Figure 3.1 The proposed framework for active and incremental learning strategy for the semantic segmentation of point clouds. 55
Figure 3.2 Variation in predicted labels within a segment. 60
Figure 3.3 An overview of the study area in Rotterdam. 63
Figure 3.4 The overview of the Amsterdam dataset. The training area is in the black box. 65
Figure 3.5 Comparison of model performance on the Rotterdam dataset with different sizes of the initial tiles (left). 68
Figure 3.6 Comparison of the model performance with three sizes of selected point cloud tiles in each iteration in the Rotterdam dataset, using point entropy function. 68
Figure 3.7 Comparison of model performance on the Amsterdam dataset with different sizes of the initial tiles. 69

Figure 3.8 Comparison of the model performance with three sizes of selected point cloud tiles in each iteration in the Amsterdam dataset, using point entropy function. 69
Figure 3.9 Comparison of model performance on the Rotterdam dataset when models are incrementally fine-tuned with only newly selected data (New data) and all available data (All data). 70
Figure 3.10 Comparison of the model performance on the Rotterdam dataset for different learning rates used in fine-tuning. 71
Figure 3.11 Mean IoU scores of baseline and active learning strategies with different query functions for the Rotterdam dataset. 72
Figure 3.12 IoU (lines) and data distribution (columns) for different classes in the Rotterdam dataset. 74
Figure 3.13 Example of tiles selected by point entropy. 75
Figure 3.14 Example of tiles selected by Mutual information. 75
Figure 3.15 Example of tiles selected by segment entropy. 76
Figure 3.16 Mean IoU scores of baseline and active learning strategies with different query functions for the Amsterdam dataset. 78
Figure 3.17 IoU (lines) and data distribution (columns) for different classes in the Amsterdam dataset. 78
Figure 3.18 Spatial distribution of selected tiles in Amsterdam. 79
Figure 3.19 Comparison of model performance on the Rotterdam dataset under different training strategies, training from scratch (TFS) and fine-tune (FT). 80
Figure 3.20 Accuracy (batch accuracy) for each update during the training. 81
Figure 4.1 The workflow of weak supervision for the semantic segmentation of ALS data based on weak subcloud labels. 95
Figure 4.2 The structure of the base network to generate PCAM and pseudo labels. 96
Figure 4.3 Pseudo label generation in MPRM (Wei et al., 2020). 97
Figure 4.4 A sample of the overlap region. 99
Figure 4.5 The relationship between the input sphere of the classification network and the subcloud for weak labels. 99
Figure 4.6 The structure of the elevation attention unit. 101
Figure 4.7 The position of the elevation unit. 101
Figure 4.8 An overview of the ISPRS benchmark dataset. 104
Figure 4.9 Classification results of our weak supervision on the ISPRS benchmark dataset (0.5% weak labels). 108
Figure 4.10 The error map of our weak supervision on the ISPRS benchmark dataset (0.5% weak labels). 109
Figure 4.11 Pseudo label accuracy and testing data accuracy when using different radii of subcloud for weak supervision. 112
Figure 4.12 Examples of classification results on the Vaihingen dataset obtained from classification networks trained with different sizes of subclouds. 112

Figure 4.13 Examples of different sampling strategies to generate subclouds. Images are 2D projections of 3D objects. 114
Figure 4.14 Quantitative results on pseudo labels over the training data and predictions on testing data when using different percentages of the overlap region for weak supervision. 116
Figure 4.15 Examples of classification results on the Vaihingen dataset obtained from classification networks trained under different overlap schemes. 117
Figure 4.16 Quantitative results on pseudo labels and predictions on testing data when the classification network without (w/o) and with (w/) elevation attention (EA). 118
Figure 4.17 Examples of classification results on the Vaihingen dataset obtained from classification networks trained with and without elevation attention (0.5% weak labels scheme) 118
Figure 4.18 Quantitative results on selected pseudo labels and predictions on test data with different values of the threshold for the ISPRS benchmark dataset..... 119
Figure 4.19 Classification results on the ISPRS testing data when using different thresholds (t) to select confident points for training.. 120
Figure 4.20 Classification results on the ISPRS testing data when using different thresholds (t) to select confident points for training.. 121
Figure 4.21 An overview of the Rotterdam dataset. 123
Figure 4.22 Qualitative results on pseudo labels for the training data of the Rotterdam data. 125
Figure 4.23 Qualitative results on pseudo labels for the training data. 126
Figure 4.24 Qualitative results on pseudo labels for the training data. 129
Figure 4.25 Qualitative results on the DFC2019 testing data. 131

List of tables

Table 2.1 Subsampling grid size and convolution radius in differet layers. 30
Table 2.2 Confusion matrix of our proposed network on ISPRS benchmark dataset.....33
Table 2.3 Quantitative comparisons between our LGENet and other models on the ISPRS benchmark dataset. 34
Table 2.4 Quantitative results (F1 scores) of hybrid KPConv with different numbers of kernel points in the 2D convolution on ISPRS benchmark dataset.....35
Table 2.5 Quantitative comparison of classification results using SegECC operations at different hybrid convolutional layers on ISPRS benchmark dataset. 37
Table 2.6 The number of segments produced with different regularization factors for the segmentation algorithm. 37
Table 2.7 Comparison of model performance on ISPRS benchmark dataset when using different regularization factors in the segmentation algorithm. 38
Table 2.8 Quantitative comparison of classification results using different segmentation methods for SegECC operation on ISPRS benchmark dataset.....39
Table 2.9 Comparison of model performance on ISPRS benchmark dataset with a different number of edges selected in SegECC operation. 40
Table 2.10 Comparison of model performance with spatial-channel attention and without spatial-channel attention.....41
Table 2.11 Quantitative comparison of classification results of PointNet++, PointNet++ with a SegECC layer and PointNet++ with a SegECC layer and a spatial-channel attention on the ISPRS benchmark dataset.....42
Table 2.12 Quantitative classification results of different models on the DFC2019 dataset. 43
Table 2.13 Quantitative comparisons between other methods and our LGENet on the DFC2019 dataset. 44
Table 3.1 Parameter configuration of multiple grouping modules in PointNet++ 66
Table 3.2 Comparison of required training time for the Rotterdam dataset when models are incrementally fine-tuned with only newly selected data (New data) and all available data (All data). 71
Table 3.3 Comparison of the computation time required for the Rotterdam dataset using different fine-tuning learning rates.....71
Table 3.4 Mean IoU scores of baseline and active learning strategies with different query functions for the Rotterdam dataset. 73

Table 3.5 Comparison of the computational time required for the Rotterdam dataset using different query functions.....76
Table 3.6 Mean IoU scores of baseline and active learning strategies with different query functions for the Amsterdam dataset.....79
Table 3.7 Comparison of the computation time required for the Amsterdam dataset using different query functions. 79
Table 3.8 Comparison of the training time required for the Rotterdam dataset using different training strategies. 82
Table 4.1 Confusion matrix of our method (0.5% weak labels) on the ISPRS benchmark testing data. 107
Table 4.2 Quantitative comparisons between full supervision and weak supervision on the ISPRS benchmark dataset. 110
Table 4.3. Quantitative results on ISPRS pseudo labels. 110
Table 4.4 Number of weak labels generated when using different subcloud radii. 111
Table 4.5 Pseudo label accuracy and testing data accuracy on the ISPRS dataset..... 111
Table 4.6 Number of labelled subclouds with different percentages of the overlap region. 114
Table 4.7 Quantitative results on pseudo labels and predictions on testing data under different overlap schemes..... 115
Table 4.8 Quantitative results on pseudo labels and predictions on testing data with and without elevation attention (0.5% weak labels scheme)..... 117
Table 4.9 Classification results on the ISPRS testing data when using different thresholds ($\boldsymbol{t}$) to select confident points for training. . 119
Table 4.10 Classification results on ISPRS testing data when using different thresholds ($\boldsymbol{t}$) to select confident points for training. . 120
Table 4.11 Classification network size and runtime comparison among different methods..... 122
Table 4.12 Segmentation network size and runtime comparison among different methods..... 122
Table 4.13 Quantitative comparisons between full supervision and weak supervision on the Rotterdam testing data. 124
Table 4.14 Quantitative results on pseudo labels on the Rotterdam dataset obtained by different strategies for pseudo label generation..... 126
Table 4.15 Predictions on the Rotterdam testing data obtained by different training strategies. 127
Table 4.16 Quantitative comparisons between full supervision and weak supervision on the DFC testing data. 128
Table 4.17 Quantitative results on pseudo labels on the DFC2019 dataset obtained by different strategies for pseudo label generation..... 129

Table 4.18 Predictions on the DFC2019 testing dataset obtained by different training strategies. 130

Chapter 1 – Introduction

1.1 Background

Point clouds are characterized by representation of objects' geometrical properties, like shape, size and orientation. They are essential data sources used to generate digital terrain models (DTM), 3D city models, landscape models and high precision maps. Products derived from point clouds can provide detailed information on natural and man-made environments, such as the number and the size of buildings and trees and the spatial distribution of different objects. They therefore have a wide range of applications in urban planning, disaster management, forest inventory and autonomous driving.

Point cloud interpretation, which targets the obtention of geometrical and semantic information, is an important step which is taken prior to the generation of the above products. For example, binary pointwise classification can be a pre-step for DTM generation, in order to separate ground and non-ground points. More categories are required for 3D city models like buildings and vegetation. With advances in Lidar techniques, Lidar point clouds are cheaper and easier to acquire so a huge amount of point cloud data is now available. However, manual interpretation of a considerable number of points can be very time-consuming, especially for 3D city models which cover large urban areas and require semantic information on multiple categories. Also, urban environments can rapidly change and slow manual interpretation may fail to keep up with those changes. Therefore, how to efficiently interpret point clouds needs to be investigated.

Finding and distinguishing 3D objects in complex urban areas can be achieved by 3D object detection and recognition. Object detection aims to find and localize objects, where bounding boxes are assigned to objects. Recognition aims to provide semantic information. These tasks are essential in many real-time applications such as autonomous driving and augmented reality. However, the limitation is that only bounding boxes are given without object boundaries which are critical for high-accuracy 3D maps.

Semantic segmentation of point clouds aims to assign every point with a semantic label and it provides more detailed semantic information on objects boundaries compared to 3D object detection and recognition. It is of importance when generating 3D products that have multiple categories and ask for detailed object geometry, such as high-accuracy 3D maps. Currently, semantic segmentation of point clouds is a very active research field and many researchers attempt to apply machine learning techniques to this task (Gerke and Xiao, 2013; Huang et al., 2016; Li et al., 2016; Weinmann et al., 2015; Xu et al., 2014; Yang et al., 2017).

1.2 Airborne laser scanning point clouds

Laser scanning is one of the approaches used to acquire point clouds. It is an active remote sensing technique in surveying communities. Laser scanners emit pulsed laser light to illuminate the target and receive reflected pulses. The distance between the scanner and the target is derived from the differences in laser return wavelengths and times (Vosselman and Maas, 2010) (Figure 1.1). The absolute 3D coordinates of a target object are calculated from the distance between the target and the sensor, the angle at which the pulse is emitted and the absolute location of the laser scanner. Laser scanners can be mounted on various platforms like aeroplanes and cars. In our research, we focus on point clouds captured through Airborne Laser Scanning (ALS) which can be performed on helicopters, fixed-wing aircraft and Unmanned Aerial Vehicles (UAVs).

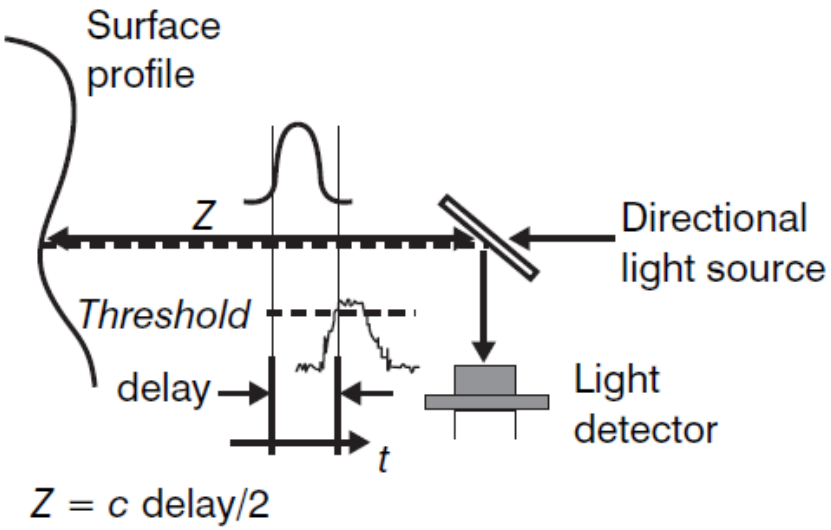


Figure 1.1 Principle of Lidar (Vosselman and Maas, 2010).

The ALS technique has two major components (Vosselman and Maas, 2010): a laser scanner system and a GPS/IMU combination. The former measures the distance to the spot hit by the laser and the latter measures the position and the orientation of the aircraft. An on-board airborne laser scanner has basically six parts namely, scanner assembly, GPS antenna, inertial measurement unit (IMU), control and data recording unit, operator laptop and flight management system. The scanner assembly is mounted on the bottom of the fuselage. It continuously emits laser pulses towards the ground and records echoes

during the flights. The GPS antenna placed on the top of the aircraft records the position of the system. The IMU records the rotation rates and acceleration data during the flight which are essential information to calculate the orientation and the position of the system with high accuracy. The combination of GPS and IMU are used to reconstruct the flight path. The control and data recording unit synchronizes time and stores all the data acquired by the components mentioned above. The operator laptop is responsible for parameter setup and system monitoring during the flight and the flight management system displays the flight plan to the pilot.

ALS point clouds are important data sources for surveying and mapping at the city level. Compared to other platforms, ALS allows quick data acquisition from the top view for large scales areas. It not only captures points from object surfaces but also penetrates through vegetation canopies to reach the ground. Therefore, DTM can be extracted from the ALS point clouds even in forestry areas.

1.3 Advantage of deep learning techniques

In recent years, machine learning has shown huge potential in automated interpretation. Machine learning algorithms can directly learn from data without using predetermined sophisticated features. Traditional classifiers such as random forest (Breiman, 2001), boosting scheme (Breiman, 1996) and support vector machine (Cortes and Vapnik, 1995) are widely used in semantic segmentation tasks. They use handcrafted features as input and produce pointwise labels. However, these predefined features are not representative enough to differentiate objects in complicated environments because they are not data-driven.

Deep learning is a specialized technique in machine learning. Its capabilities were first proven in 'Large Scale Visual Recognition Challenge' (Russakovsky et al., 2015). It achieved record-breaking results in both classification and localization tasks in the world's greatest computer vision competition. From then on, as an efficient state-of-the-art approach to extracting information from visual data, deep learning has continued to show its powerful abilities in object recognition and semantic segmentation from both images and point clouds. The huge success of deep learning is due to its ability to learning features from different levels based on data instead of using the predefined features in traditional machine learning methods.

Convolutional Neural Network (CNN) is one of the most important deep learning algorithms in image related tasks. The network consists of a sequence of convolutional layers and pooling layers. Shallow

convolutional layers have relatively small receptive fields and can capture local features, while deep layers have larger receptive fields and can extract more abstract features at higher levels. These higher-level features are more robust to variance in size, location and orientation of objects, which improve model performance.

Deep learning also shows its power in point cloud processing. Unlike images with regular grid style, points are sparsely distributed without any permutation. To take advantage of CNNs, some methods try to convert point clouds into the grid style in order to make the data adapt to CNN networks. They project 3D point clouds to 2D images from different views (Kalogerakis et al., 2017) or attempt to convert point clouds into well-aligned 3D grids (Maturana and Scherer, 2015). Recently, researchers have paid more attention to how raw points can be taken as the network input without any conversion that may cause artefacts. PointNet (Qi et al., 2017a) is the prior work that constructs the network by a set of multi-layer perceptrons and first allows the network to learn from sparely and regularly distributed points. Inspired by the image convolutions, many researchers design new convolutions (Li et al., 2018; Thomas et al., 2019) for point clouds that not only directly take points as the input but are also more effective in learning geometrical features from point clouds compared to conventional CNNs.

1.4 Research gap

For point clouds, inherent structures of objects are the keys to distinguishing different categories. Those structures can be captured at multiple scales. However, how to enhance the learning of those representative features from ALS point clouds, with the intention to boost the network performance, is still an open topic to be researched. In conventional deep network training for semantic segmentation, the training data needs to be fully annotated. Even though pointwise labels on the whole training data are helpful to avoid overfitting and to have better generalization to the testing data, labelling 3D points is much more difficult than image annotation and 3D annotation is tedious and time-consuming. Thus, how to maintain the network performance with fewer annotation efforts on the training datasets, for instance annotating fewer but informative samples and using cheaper labels, are required to be investigated.

1.5 Research objectives

This Ph.D. thesis investigates the semantic segmentation of ALS point clouds based on deep learning algorithms. We first explore how to learn representative features from ALS point clouds and then focus on how

to reduce the manual labelling efforts to train a deep learning model for semantic segmentation. For the training, we investigate active learning to select and annotate only informative points and weak supervision to annotate only weak labels for the pointwise prediction task. Figure 1.2 demonstrates the structure of this research. The Details for each objective are as follows:

1). Semantic segmentation of ALS point clouds. The major objective is to allow the network to learn representative features from ALS data and involve different levels of neighbouring information to extract pointwise geometrical features. Firstly, local geometrical features for ALS point clouds are extracted by a designed feature extractor. Then contextual information is explored at both object and global levels.

2). Active and incremental learning for semantic segmentation of ALS point clouds. The primary objective is to reduce the required annotation efforts for the training of deep learning models by selecting and annotating the most informative ALS point clouds. Under the active learning strategy, how to effectively select informative samples for ALS datasets is explored. As active learning requires model training every time newly annotated samples are involved, how to effectively train the models is explored.

3). Weak supervision on semantic segmentation of ALS point clouds. This objective aims to alleviate the annotation efforts for the deep learning training through the use of weak subcloud labels instead of pointwise ground truth for ALS datasets. There are two basic components for this objective, the classification network and the segmentation network. We first explore how to use weak subcloud labels to train a classification network that can produce pointwise pseudo labels on the training data. Then we investigate how to make use of the produced pseudo labels to train a segmentation network that gives predictions on the testing data.

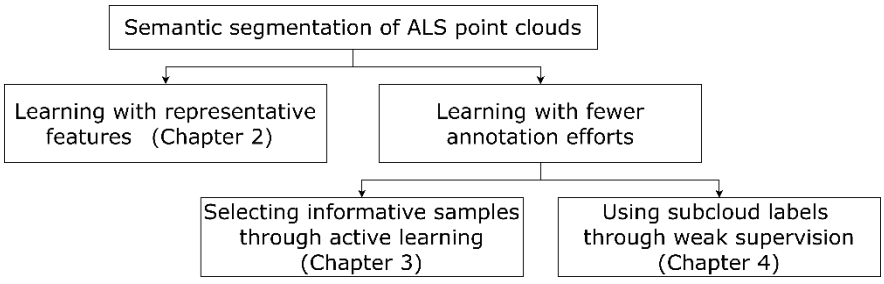


Figure 1.2 The structure of the research objectives in this thesis.

1.6 Outline

This thesis is composed of five chapters. Objectives 1, 2, 3 mentioned above are addressed in Chapters 2, 3, 4. Each of these chapters is based on a journal paper which is listed on the first page of each chapter. The literature review in these chapters may have some overlap. Details of each chapter are as follows:

Chapter 1 presents the background and introduces the ALS point clouds and the deep learning techniques. Then the research gap is explained and the primary objectives of this research are listed. Finally, an outline is given to show the structure of this thesis.

Chapter 2 investigates how to learn representative features from ALS point clouds for semantic segmentation using the deep learning algorithm. We first adapt the KPConv to a 2D-3D convolution block to extract local representative features for ALS point clouds. Then we propose a segment-based edge conditioned convolution to encode context at the object level. Finally, we exploit the global contextual information by adding a spatial-channel attention module at the end of the network.

Chapter 3 explores how to select informative training samples through an active learning strategy in order to effectively reduce the required annotation efforts for the training of deep learning models for the semantic segmentation of ALS point clouds. Different evaluation metrics are compared and contrasted after which the optimal one is chosen for the semantic segmentation of ALS point clouds. The incremental learning is also introduced to the active learning framework for the purpose of effectively reducing the training efforts.

Chapter 4 addresses the training of deep learning networks with weak subcloud labels. We first improve the classification network in order to produce better pointwise pseudo labels on the training data by exploiting the semantic heterogeneity within a subcloud and encoding more representative features by an elevation attention module. Then, a supervised contrastive loss is adopted to unravel the underlying correlations of class-specific features for the segmentation network trained on the produced pointwise pseudo labels.

Chapter 5 synthesizes the research. It draws conclusions on this research and makes recommendations for future research.

Chapter 2 – Local and Global Encoder Network for Semantic Segmentation of Airborne Laser Scanning Point Clouds [1]

[1] This chapter is based on:

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2021. Local and global encoder network for semantic segmentation of Airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 176, 151–168.

Abstract

Interpretation of Airborne Laser Scanning (ALS) point clouds is a critical procedure for producing various geo-information products like 3D city models, digital terrain models and land use maps. In this chapter, we present a local and global encoder network (LGENet) for semantic segmentation of ALS point clouds. Adapting the KPConv network, we first extract features by both 2D and 3D point convolutions to allow the network to learn more representative local geometry. Then global encoders are used in the network to exploit contextual information at the object and point level. We design a segment-based Edge Conditioned Convolution to encode the global context between segments. We apply a spatial-channel attention module at the end of the network, which not only captures the global interdependencies between points but also models interactions between channels. We evaluate our method on two ALS datasets namely, the ISPRS benchmark dataset and DCF2019 dataset. For the ISPRS benchmark dataset, our model achieves state-of-the-art results with an overall accuracy of 0.845 and an average F1 score of 0.737. With regards to the DFC2019 dataset, our proposed network achieves an overall accuracy of 0.984 and an average F1 score of 0.834.

2.1 Introduction

With the advanced techniques of light detection and ranging (LiDAR) systems, point clouds are more easily obtained in various scenes. Airborne laser scanning (ALS) point clouds have become an essential type of data in the generation processes of digital terrain models (DTM) (Chen et al., 2017), landscape models (Murtha et al., 2018), 3D city models (Lin et al., 2018) and land use maps (Meng et al., 2012). These point cloud based products are required in many disciplines, like urban planning (Murgante et al., 2009), land administration (Lemmen et al., 2015), forest inventory (Wallace et al., 2012), tourism (Cooper et al., 2013) and disaster management (Shen et al., 2010). The interpretation of ALS point clouds is a prerequisite for their use in these applications. One of the interpretation methods is semantic segmentation which assigns a semantic label to each point in the dataset. Manually labelling every point is quite time-consuming, especially for large urban areas. Thus, machine learning techniques are developed to automate the interpretation process (Vosselman and Maas, 2010).

Machine learning approaches used for 3D scene understanding traditionally focused on extracting representative handcrafted features to describe local geometry (Lin et al., 2014; Weinmann et al., 2013) and training different discriminative classifiers to produce pointwise labels like Supported Vector Machine (SVM) (Lodha et al., 2006), AdaBoost (Lodha et al., 2007), Random Forests (RF) (Chehata et al., 2009), Gaussian Mixture Model (GMM) (Weinmann et al., 2014) and Artificial Neural Networks (ANN) (Xu et al., 2014). The involvement of contextual information between points has been proven to be effective in improving semantic segmentation results and this can be achieved by using graphical models such as Conditional Random Field (CRF) (Niemeyer et al., 2016, 2011; Vosselman et al., 2017). However, in these methods, low dimensional handcrafted features are not representative to distinguish all categories in the dataset especially for the ALS point clouds acquired over complicated scenes where objects are largely different in size.

Recently, deep learning methods have shown their powerful abilities in object recognition and semantic segmentation from images. The huge success of deep learning is due to learning features from different levels based on data instead of using the predefined features in traditional machine learning methods. Inspired by the success of deep learning in image related tasks, many deep learning based approaches for 3D interpretation tasks are proposed, like image-based methods (Boulch et al., 2018; Kalogerakis et al., 2017), voxel-based methods (Maturana and Scherer, 2015; Tchapmi et al., 2017) and point-based

methods (Li et al., 2018; Qi et al., 2017a; Thomas et al., 2019). With regards to ALS point clouds, some researchers convert point clouds into sets of images (Hu and Yuan, 2016; Yang et al., 2017; Zhao et al., 2018) or 3D voxel grids (Schmohl and Sörgel, 2019). Others design networks (Arief et al., 2019; Li et al., 2020; Winiwarter et al., 2019; Yousefhussien et al., 2018) that can directly consume ALS point clouds and learn more representative features with less information loss.

Global contextual cues have proven that they can further improve the results from deep learning methods for computer vision tasks. Some approaches use fully connected CRF to enforce global consistency and refine semantic predictions on images (Zheng et al., 2015) and point clouds (Tchapmi et al., 2017). Motivated by the self-attention module proposed by Vaswani et al. (2017) for machine translation, various other approaches adapt this concept for computer vision tasks like semantic segmentation of images (Wang et al., 2018) and point clouds (Feng et al., 2020). Nevertheless, these methods explore dependencies between pixels or points, ignoring relationships between objects which are informative for large scale complex outdoor scenes. Super point graph (SPG) (Landrieu and Simonovsky, 2018) only assigns labels to segments and incorrect segmentation causes errors in the final pointwise predictions. Therefore, it is still challenging to make use of global context at both point and object levels for large scales ALS data.

In this chapter, we propose a novel 3D convolutional network, a local and global encoder network (LGENet), that can embed more representative features for ALS data and exploit global context at both object and point levels. Considering that the variance of ALS point cloud coordinates is larger in the XY plane than along the Z-axis, we first enhance the representativeness of features obtained by 3D convolutions by adding 2D convolutions in order to pay more attention to the point distribution on the XY plane. Next, motivated by SPG (Landrieu and Simonovsky, 2018), we encode global interdependencies between segments by segment-based Edge Conditioned Convolution (SegECC). Segments are obtained from the unsupervised algorithms before the training and trainable edge conditioned convolutions are applied to capture the spatial dependencies between objects. This operation can be inserted after any convolutional layers in the network. Finally, a spatial-channel attention is introduced to semantic segmentation of point clouds and placed at the end of the network to capture long-range interactions between points and dependencies between channels. The major **contributions** of this chapter are listed as follows:

1) We propose a hybrid block that combines features extracted from both 2D and 3D convolutions. 2D convolutions are introduced to allow the network to learn representative features for point clouds primarily distributed in horizontal dimensions.
2) To capture global spatial dependencies at the object level, we design a SegECC operation that constructs graphs on segments and exploits the relationships among objects. Segment features are then concatenated to pointwise features to allow the network to adaptively encode local-global features.
3) To make use of spatial and channel dependencies, a spatial-channel attention is modified for semantic segmentation of ALS point clouds. The spatial attention learns the global interactions between points and channel-wise attention enhances the discriminability of learned features for different semantic categories.

The remainder of the chapter is structured as follows. In Section 2.2, we review related traditional methods and recent deep learning methods in semantic segmentation of point clouds. Section 2.3 introduces the hybrid convolution, SegECC and spatial-channel attention designed in our network. In Section 2.4, we show our results on the ISPRS benchmark dataset (Niemeyer et al., 2014) and compare LGENet against other state-of-the-art models. Extensive ablation experiments are carried out on the ISPRS benchmark dataset (Niemeyer et al., 2014) to evaluate our proposed method. We also test our model with the DFC2019 dataset (Bosch et al., 2019). Section 2.5 concludes this chapter.

2.2 Related work

2.2.1 Traditional methods

Traditional machine learning methods for classifying point clouds are generally divided into two steps, extracting handcrafted features and training discriminative classifiers. For semantic segmentation of ALS datasets, many researchers define features that describe local geometry. Lin et al., (2014) propose a method to compute 'eigenfeatures' from the covariance matrix of local neighbouring points to characterize local point distributions of ALS point clouds, e.g. planarity, sphericity, linearity. Then a Support Vector Machine (SVM) is used to classify point clouds. Weinmann et al. (2013) evaluate 21 geometric features including 8 'eigenfeatures' derived from optimal local neighbours and a set of 2D geometric features to describe the local characteristics. These features are then tested with four classifiers, namely, nearest neighbour, k Nearest Neighbour, Naive Bayesian and SVM. However, these methods take each point's local

geometry independently for pointwise prediction and ignore the spatial dependencies, resulting in prediction noises and label inconsistency.

The above issues can be addressed by taking advantage of the contextual information. An important statistical method to model the context is probabilistic graphical models, such as Conditional Random Fields (CRFs). Niemeyer et al. (2014, 2011) propose a pointwise classification method using CRF for ALS datasets. Unary potential is the pointwise probability distribution over classes produced by a learned classifier. Pairwise potential, revealing prominent relations between the data and object classes, is also learned during the training. Although this CRF based method gives rise to smoother results and improves class-specific accuracy, especially for classes with fewer instances, the pairwise CRF still takes interactions at a very local level into account and cannot avoid incorrect labelling to isolated point clusters. A longer-range of interactions between points is a possible solution. Xiong et al. (2011) propose a sequence of stacked classification procedures. They propagate pointwise classification to segments and then consider contextual information according to the segment-based results for the final pointwise prediction. Niemeyer et al. (2016) propose a two-layer hierarchical higher order CRF for semantic segmentation of ALS data based on the robust P^n Potts model (Kohli et al., 2009). The first layer, operating on points, takes both handcrafted geometric features and relations between points to produce pointwise labelling. In the second layer, nodes are represented by segments that are generated by a variant of the region growing algorithm, so that interactions between objects are considered. However, these methods need to extract handcrafted features before the training which are not representative for multiple categories in point clouds acquired from complex scenes.

2.2.2 Deep learning methods

The effectiveness of deep learning approaches has been proven in recent research and the idea of deep learning has been applied to point clouds interpretation.

As CNNs are capable of learning highly representative features in many image processing tasks, many strategies are proposed to adjust classical 2D image deep neural networks to 3D point clouds. One branch of methods is based on the concept of converting the unordered and irregularly distributed point clouds into rasterized 2D representations which are the input of the CNNs. For example, Kalogerakis et al. (2017) propose a fully convolutional network, ShapePFCN, for 3D part segmentation. The network input is rendered images of 3D shapes captured from different views. In addition to ShapePFCN, this projection-based method has been extended to the

semantic segmentation of large-scale point clouds with complicated scenes. Boulch et al. (2018) generate images containing geometric features obtained from the depth and RGB information. Then fully convolutional networks produce pixel-wise labels and these labels are converted into 3D space through a fast back-projection. Nevertheless, self-occlusion is difficult to avoid during the projection, especially for complicated outdoor scenes. For semantic segmentation of large-scale ALS data, numerous methods convert 3D point clouds into 2D rasterized features from the top view in order to pass the data through image based CNNs. Hu and Yuan (2016) conduct the ground points labelling of ALS data by assigning simple attributes to each pixel like minimum, maximum and mean of the height within each grid cell. Similarly, Yang et al. (2017) also apply 2D grids to 3D point clouds but they assign more full-waveform and geometric features to each 2D grid. Zhao et al. (2018) produce multi-scale contextual images that represent point set features like height, intensity and roughness. These methods of processing ALS point clouds require complicated features to be produced before the network training. Many pre-calculated features can be redundant and require large memory for data processing during the training. Also, only extracting features from projected point clouds on two 2D spaces leads to information loss along the third dimension.

Volumetric approaches that voxelize unordered point clouds into regular 3D grids are alternatives to processing point clouds in order to adapt to deep neural networks. Maturana and Scherer (2015) convert the sparsely distributed point clouds into $32 \times 32 \times 32$ binary occupancy grids where each voxel is categorized into occupied and unoccupied. Then voxelized point clouds are processed by 3D convolutions for fast object detection. 3DShapeNet (Wu et al., 2015) also uses binary 3D voxel grids as the network input for object recognition and shape completion. SegCloud (Tchapmi et al., 2017) is a 3D CNN that generates coarse down-sampled labels for each voxel. Then pointwise labels are obtained by transferring the voxel labels back points through trilinear interpolation. Concerning ALS datasets, Schmohl and Sörgel (2019) take voxelized ALS point clouds as the input of sparse submanifold convolutional networks (SSCNs). The voxelization unavoidably leads to information loss and causes artifacts. These disadvantages negatively impact the learning of representative 3D features. In addition, a large number of unoccupied grids stored in voxel structures result in high memory requirements.

Recent research focuses on how to make the deep neural network directly consume point clouds to minimize information loss. PointNet, a deep learning network designed by Qi et al. (2017), can directly

process unstructured points without any rasterization or voxelization and it achieves compelling performance on a series of point cloud related tasks, like object classification, part segmentation and semantic segmentation. PointNet learns representative point set features by Multilayer Perceptron (MLP) layers. Spatial transformers which produce transformation matrices are also auxiliary learned to align input point clouds to a canonical space and improve the robustness to geometric transformations. The key limitation is that PointNet treats each point independently. It can only encode each point individually and aggregate point features into one global representation, failing in capturing local structures. To address the above issues, Qi et al. (2017b) present a hierarchical deep network called PointNet++. It consists of a sequence of set abstract modules that progressively capture geometric features in wider and wider local regions. Instead of using farthest point sampling applied in PointNet++, RandLA-Net (Hu et al., 2020) is built on random sampling. Local feature aggregation modules are designed to capture complex local geometry. The feature aggregators first use MLPs to encode relative point positions in a local neighbourhood and then attentively pool those encoded features to the central point. Wang et al. (2020) innovatively construct a hierarchical network called WreathProdNet. It achieves the state-of-the-art on some public 3D datasets. The network is based on the symmetries of hierarchical structures which are expressed by the wreath product of the group. JSIS3D (Pham et al., 2019) is a joint semantic-instance segmentation network built on PointNet. Semantic labels and instance labels are jointly optimized by a multi-value conditional random field.

As 2D convolutional kernels have shown their effectiveness in capturing relationships in local neighbourhoods, deep neural networks based on the concept of 3D convolutions are proposed to extract representative features from local structures of point clouds. Unlike Voxnet (Maturana and Scherer, 2015) which only takes a grid-style input, these networks are able to directly process irregularly distributed point clouds and some of them define convolutional function over continuous 3D space where weights of points within a local neighbourhood depend on their spatial distributions around the central point. For example, Kernel Point Convolutions (KPConv) proposed by Thomas et al. (2019) are defined over continuous space. Linear correlation between point positions and kernel point positions defines weights of points to different areas inside convolutional kernels. Kernel point positions are learnable and this helps convolution kernels to adapt to local structures in a better way. Following Thomas et al. (2019), Varney et al. (2020) introduce spatial and channel attention to KPConv in order to capture more descriptive features. Instead of constructing a U shape network used in Thomas et al. (2019), Varney et al. (2020) construct a Pyramid Point network to

densely connect all convolutional layers. InterpConv (Mao et al., 2019), PointConv (Wu et al., 2019), SpiderCNN (Xu et al., 2018), Flex-Convolution (Groh et al., 2019) and ConvPoint (Boulch, 2020) are also 3D convolutional operators defined over continuous space to capture local contextual information. The operators can also be defined over discrete space. For example, FKAConv (Boulch et al., 2020) is designed to learn a transformation of irregularly distributed input points in order to align them with the grid-style kernel.

With regards to semantic segmentation of ALS point clouds, Yousefhussien et al. (2018) modify the PointNet and make the network learn from more input features which consist of XYZ coordinates and corresponding radiometric features extracted from IR-R-G imagery. AlsNet based on PointNet++ is proposed by Winiwarter et al. (2019). A batching framework is introduced to allow the network to process large scale point clouds. Lin et al. (2020) also apply PointNet++ to large scale ALS data and an active and incremental learning strategy is proposed to make the training more efficient. An Atrous XCRF network is designed by Arief et al. (2019) to avoid overfitting during deep network training (eg. PointCNN) for ALS datasets that are small in size. Wen et al. (2020) first project 3D points to a horizontal plane and then use a directionally constrained point convolution to encode neighbouring orientation information in ALS data. Li et al. (2020) apply a dense hierarchical architecture with geometry-aware convolutions and an elevation-attention module to fully embed characteristics of ALS point clouds.

Exploiting global contextual information is also researched in 3D deep neural networks for semantic segmentation of point clouds. Tchapmi et al. (2017) use a fully connected conditional random field (FC-CRF) at the end of the 3D CNN to exploit long-range interactions among points. The FC-CRF is implemented as a differentiable Recurrent Neural Network. This formulation allows the joint training of 3D CNN and 3D FC-CRF. Landrieu and Simonovsky (2018) first partition large point clouds into geometrical homogenous point sets called superpoints and then apply graph convolutions to the graph constructed by superpoints. Gated Recurrent Units are implemented to exploit the long-range relationships among superpoints. Huang et al. (2020) achieve global optimization for semantic segmentation of ALS point clouds through the Markov random fields algorithm which is a post-processing to refine initial classification results.

2.2.3 Attention models

Attention can be used as a tool to pay more attention to the most informative signals during data processing and attention models have

been widely used in natural language processing and computer vision tasks. Recently, the attention mechanism has shown its potential in encoding global contextual information. Vaswani et al. (2017) propose a self-attention module for machine translation. The idea is to encode the context at one position in a sequence by calculating a weighted average of embeddings at all positions. With regards to computer vision tasks, Wang et al. (2018) propose a non-local operation in order to capture long-range spatial dependencies. The non-local operation produces attention maps by computing the correlation between all possible point pairs in the feature space and those attention maps guide the aggregation of spatial contextual information. Apart from modelling spatial dependencies, channel-wise relationships are also exploited by attention mechanisms in order to enhance the representative power of deep learning models. For example, Hu et al. (2018) design squeeze-and-excitation blocks which firstly squeeze spatial features into a channel descriptor for each channel and then recalibrate channel-wise features by modelling channel-wise interdependencies. DANet (Fu et al., 2018) takes advantage of both spatial and channel-wise attentions. Their outputs are fused at the end of networks to boost feature representation, contributing to more precise predictions. Concerning the semantic point cloud segmentation, Feng et al. (2020) insert several pointwise spatial attention modules into deep neural networks to make use of interdependencies among all points regardless of their distance. The effectiveness of the pointwise spatial attention has been proven on the ShapeNet (Wu et al., 2015) and two indoor datasets namely, ScanNet (Dai et al., 2017) and S3DIS (Armeni et al., 2016). It is also valuable to explore how to take advantage of attention modules to boost feature representation of networks by modelling both spatial and channel-wise interdependencies for the semantic segmentation of ALS datasets.

2.3 Method

We first describe the design of 2D convolutions and how 2D and 3D convolutions work together in Section 2.3.1. Then we introduce the SegECC operation that encodes the contextual information at the object level in Section 2.3.2. Section 2.3.3 explains how the spatial-channel attention is adjusted to 3D point clouds. Finally, the architecture of LGENet is presented in Section 2.3.4.

2.3.1 Hybrid convolution block

KPConv (Thomas et al., 2019) is a 3D convolutional kernel whose domain is a spherical 3D space. It has a deformable version that adapts to local geometry in order to enhance the representation of features. However, Thomas et al. (2019) suggest that rigid convolutions perform

better than deformable ones on scenes that lack of diversity. As a majority of objects in ALS datasets are buildings, ground and vegetation, pedestrians and road furniture are less likely to be observed, we use rigid convolutions in our experiments. If it is not specified, KPConv represents rigid KPConv in the following chapter. In order to extract more representative features for ALS point clouds, a 2D variant of KPConv is applied and is incorporated with 3D KPConv forming a hybrid block.

As ALS point clouds are acquired by airborne LiDAR equipment from a top view and most semantic objects in urban scenes are horizontally distributed on the ground, point cloud variance in the vertical direction (z coordinates) is much less than that in the horizontal plane. Due to this characteristic, 2D convolutions are applied to learn more representative features for urban objects from the point distribution on the horizontal plane. Their effectiveness has been proven by various previous works (Wen et al., 2020; Yang et al., 2017; Zhao et al., 2018), in which point clouds are projected to the horizontal plane and 2D CNN is applied to extract features and predict pointwise semantic labels. In the following section, the mechanism of KPConv is reviewed according to Thomas et al. (2019). Thereafter, how the 2D KPConv works and how the 2D and 3D KPConv are combined are explained.

Given a point cloud $\mathcal{P} \in \mathbb{R}^{N\times 3}$ and the corresponding feature $\mathcal{F} \in \mathbb{R}^{N\times C_1}$, at a point $p \in \mathbb{R}^3$, the point convolution of $\mathcal{F}$ taken by the kernel function g is written as follows:

$$(\mathcal{F} * g)(p) = \sum_{p_i \in N_p} g(p_i - p) f_i \tag{2.1}$$

where N_p is a set of neighbours around p within a fixed radius $r \in \mathbb{R}$, $N_p = \{p_i \in \mathcal{P} \mid \|p_i - p\| \leq r\}$. p_i is one of the neighbours for p in point set $\mathcal{P}$ and its corresponding feature is $f_i \in \mathbb{R}^{C_{in}}$, where C_{in} is the number of input feature channels. For simplicity, we set the input of function g as $x_i = p_i - p$, and $\{x_i \in \mathbb{R}^3 \mid \|x_i\| \leq r, i \in 1, 2, \dots, N'\} \subset D_r^3$. x_i is the relative position of neighbouring points to the central point p and N' is the number of neighbours. D_r^3 represents the domain of g for 3D KPConv, which is a 3D ball space centred on p with a radius r. Similar to image convolutional kernels, the kernel function g has different weights to different parts inside the kernel domain. Different areas in D_r^3 are localized by a set of kernel points $\{\tilde{p}_k \in \mathbb{R}^3 | k < K\} \subset D_r^3$. $\tilde{p}_k$ is a 3D position in D_r^3 and K is the number of kernel points for the kernel function g. The corresponding weight matrices of the kernel points are denoted as $\{W_{pk} \,|k < K\} \subset \mathbb{R}^{C_1 \times C_2}$, mapping features from dimension C_1 to C_2. The kernel function g for any input $x_i \in D_r^3$ is defined as:

$$g(x_i) = \sum_{k<K} h(x_i, \tilde{p}_k) W_{pk} \tag{2.2}$$

where h is a linear correlation between $\tilde{p}_k$ and x_i. h is larger when x_i is close to the k^{th} kernel point $\tilde{p}_k$.

The 2D kernel function g_{2d} is quite similar to the 3D one g, except N_q and kernel point position $\tilde{q}_k$ defined differently. The point convolution taken by the 2D kernel function g_{2d} at $q \in \mathbb{R}^2$, the projection of the point $p \in \mathbb{R}^3$ on the XY plane, is defined as:

$$(\mathcal{F} * g_{2d})(q) = \sum_{q_i \in N_q} g_{2d}(q_i - q) f_i \tag{2.3}$$

Instead of searching neighbours among projected 2D points, N_q is the 2D projection of N_p. Therefore, N_q and N_p have the same number of points N'. We define the input of function g_{2d} as $y_i = q_i - q$ and $\{y_i \in \mathbb{R}^2 \mid \|y_i\| \leq r, i \in 1, 2, \dots, N'\} \subset D_r^2$. y_i is the relative position of q_i to the central point q and D_r^2 is the domain of g_{2d} which is a 2D circular surface centred on q with a radius r. The 2D kernel points in D_r^2 are written as $\{\tilde{q}_o \in \mathbb{R}^2 \mid o < O\} \subset D_r^2$. $\tilde{q}_o$ is a 2D coordinate in D_r^2 and O is the number of kernel points for the 2D kernel function g_{2d}. The corresponding weight matrices of the 2D kernel points are notated as $\{W_{qo} \mid o < O\} \subset \mathbb{R}^{C_1 \times C_2}$. The 2D kernel function g_{2d} is defined as:

$$g_{2d}(y_i) = \sum_{o<O} h(y_i, \tilde{q}_o) W_{qo} \tag{2.4}$$

The distributions of 3D and 2D kernel points are shown in Figure 2.1. It can be seen that in the 3D kernel, top and bottom kernel points are far away from the object surface so that those points have little to no contribution in describing the local geometry. However, when the distance between a point and a kernel point is only assessed in 2D, most kernel points will have some nearby points on object surfaces. In this way, more kernel points contribute to the feature extraction, leading to more representative features on object surfaces.

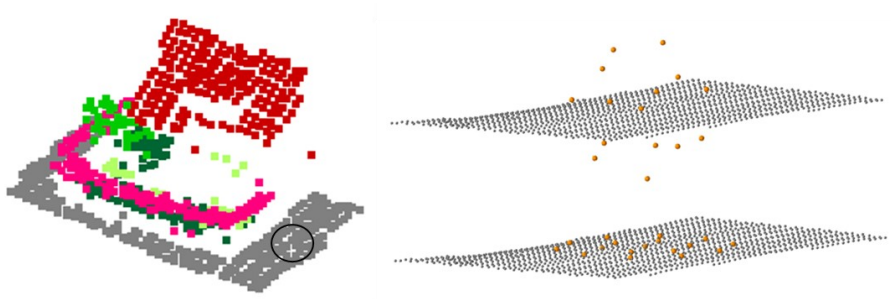


Figure 2.1 Illustration of kernel points distribution for 2D and 3D KPConv. Left: an example of ISPRS benchmark dataset. Right top: the perspective view of 3D kernel points. Right bottom: the perspective view of 2D kernel points. Kernel points are shown in orange.

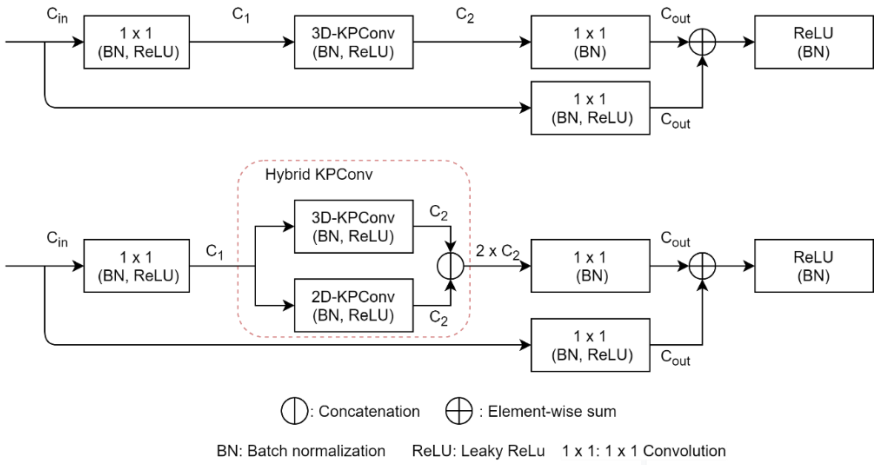


Figure 2.2 The convolutional block used in Thomas et al. (2019) (top) and the hybrid 2D-3D block used in this chapter (bottom). The hybrid block inherits the ResNet connections from the original one. Instead of simply passing through a 3D-KPconv, features are fed to both 2D and 3D KPConvs and outputs of two convolutions are concatenated for the 1×1 convolution.

The hybrid KPConv block used in the network is shown in Figure 2.2. It inherits the ResNet connections from the original block (Thomas et al., 2019) shown in the top block of Figure 2.2. The single 3D-KPConv in the original block is replaced with a 3D-KPConv and a 2D-KPConv. The hybrid block first maps the input feature from dimension C_{in} to dimension C_1 through a 1×1 convolution layer followed by a batch normalization layer and ReLu**.** The input features are also transformed by another 1×1 convolutional layer to obtain $\mathcal{F}' \in \mathbb{R}^{N \times C_{out}}$. The mapped features $\mathcal{F} \in \mathbb{R}^{N \times C_1}$ are taken as the input of 2D and 3D kernel function

whose output feature dimensions are C_2. Then, output features are concatenated to form the feature $\mathcal{F}'' \in \mathbb{R}^{N \times C_3}$, where $C_3 = 2 \times C_2$. $\mathcal{F}'' \in \mathbb{R}^{N \times C_3}$ is then transformed to be $\mathcal{F}''' \in \mathbb{R}^{N \times C_{out}}$ by a 1×1 convolutional layer. Finally, an elementwise summation between $\mathcal{F}'''$ and $\mathcal{F}'$ is implemented to form a residual block and produce final output features.

2.3.2 SegECC

The pointwise features obtained by hybrid KPConv layers are only representative for local geometry because each convolutional layer only has a local receptive field and pointwise features cannot encode information outside the local region as well as relationships between objects. This is insufficient to explore the inherent structures of large objects and the interactions between objects. Lack of this global context limits the network performance on pointwise prediction for outdoor scenes in ALS point clouds. To achieve better performance, spatial dependencies at the object level from a global perspective should be exploited and integrated with local geometrical features. Inspired by SPG (Landrieu and Simonovsky, 2018), we construct graphs on segments that consist of geometrical homogenous points to capture the relationships among objects. By combining segment features and pointwise features, the network adaptively encodes local-global features, thus achieving better semantic predictions on ALS datasets. The following paragraphs explain how global context is explored at the segment level and how it is aggregated with local features.

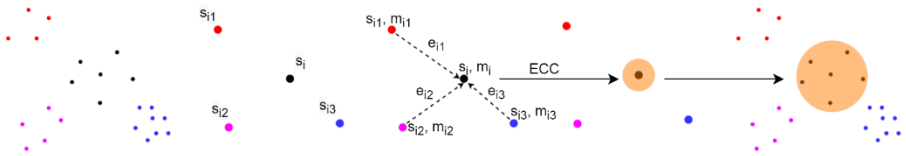


Figure 2.3 Steps in SegECC to obtain segment embeddings using edged conditioned convolutions (ECC). Different colors represent different segment labels.

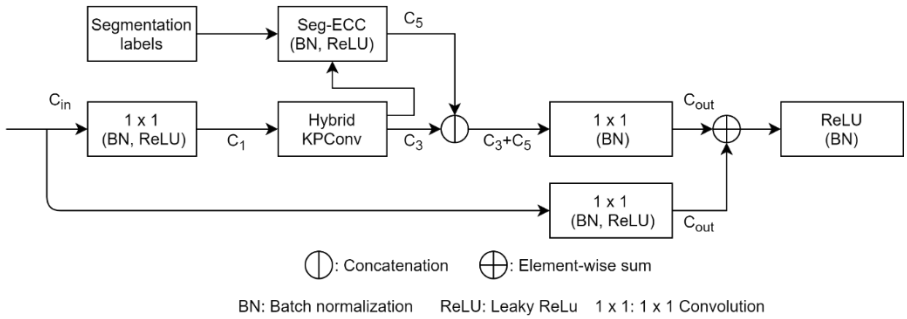


Figure 2.4 The structure of the hybrid-SegECC block.

Figure 2.3 illustrates the process of the segment-based Edge Conditioned Convolution (SegECC) to obtain global embeddings. Firstly, point clouds are partitioned into segments by an unsupervised algorithm, L_0-cut pursuit proposed by Landrieu and Obozinski (2017). The segmentation is performed before the training and is based on predefined geometrical features and intensity. Unlike DNN (Simonovsky and Komodakis, 2017) which dynamically clusters points according to updated features during the training, we use a fixed graph structure in our network and all segment labels are inherited from the initial segmentation. This fixed structure is more computationally efficient because it does not search KNN neighbours in high dimensional feature space for every training iteration. Experimental results in Section 2.4.1.8.2 show the effectiveness of the fixed segment labels. Next, pointwise features obtained from hybrid KPConv $\mathcal{F}'' \in \mathbb{R}^{N \times C_3}$ are aggregated to node features $M \in \mathbb{R}^{N_{seg} \times C_3}$ according to segment labels where N_{seg} is the number of segments within the scene. Within a segment, node features are calculated as averages of features over all segment points. For each node s_i, a graph is constructed with all other nodes $\{s_{ij} \mid j < (N_{seg} - 1)\}$ in the scene. The features at the central node s_i are represented by $m_i \in \mathbb{R}^{C_3}$ and the features of s_{ij} are represented by $m_{ij} \in \mathbb{R}^{C_3}$. Edge features are represented by $e_{ij} = m_i - m_{ij}$. Then Edge-Conditioned Convolution (ECC) (Simonovsky and Komodakis, 2017) is used to capture the contextual information among different segments. It can dynamically generate filtering weights are according to e_{ij} and deal with a flexible number of neighbours. The calculation of ECC is shown as the following:

$$m_i' = \frac{1}{N_{seg} - 1} \sum_{j < (N_{seg} - 1)} \Theta(e_{ij}, W_e) m_{ij} \tag{2.5}$$

W_e are learnable parameters in the multi-layer perceptron Θ. Edge features e_{ij} are processed by Θ to produce a weight matrix and a

matrix-vector multiplication is performed between the weight matrix and neighbouring node features m_{ij}. Adaptively weighted m_{ij} are aggregated by the calculation of the mean. Finally, updated node features $M' \in \mathbb{R}^{N_{seg} \times C_4}$ are directly mapped to each point as $\mathcal{F}_{ecc} \in \mathbb{R}^{N \times C_4}$. Then $\mathcal{F}_{ecc}$ are concatenated with pointwise features $\mathcal{F}''$ generated from KPconv and the concatenated features are mapped to dimension C_5 by a 1×1 convolutional layer. Figure 2.4 demonstrates how SegECC operation is inserted into a hybrid convolution block. The input of the SegECC is the feature obtained from the hybrid KPConv $\mathcal{F}''$ and its output is $\mathcal{F}'_{ecc} \in \mathbb{R}^{N \times C_5}$. $\mathcal{F}''$ and $\mathcal{F}'_{ecc}$ are concatenated for following operations.

2.3.3 Spatial-channel attention

Hybrid KPConv and SegECC are proposed to extract representative features at point and object levels. However, it is also necessary to consider global information when determining the semantic label for each point. In semantic segmentation, two points can be the same category even if they are spatially far away. Considering the correlation between these two points in feature space can mutually improve the prediction accuracy. Also, for high dimensional features, dependencies between channels exist, which enhance the feature discriminability for different semantic classes. Following the dual attention proposed by Fu et al. (2018) for image semantic segmentation, the spatial-channel attention is proposed for semantic segmentation of ALS point clouds.

In order to model the relationship between any two members in a point cloud, the spatial-attention Module is applied to adaptively aggregate pointwise features according to their correlations. The spatial attention module in Fu et al. (2018) is built on the self-attention mechanism proposed by Vaswani et al. (2017) for machine translation. According to Vaswani et al. (2017), the attention function maps queries and key-value pairs to outputs. The outputs are calculated as the weighted sum of the values and the corresponding weights can be obtained from pairwise functions between the queries and their corresponding keys which represent query-key relationships. Fu et al. (2018) adapt this self-attention concept to image semantic segmentation tasks. The input feature is projected to different feature subspaces through different learnable fully connected layers in order to construct queries, keys and values in the attention function. The outputs of the attention function are feature-enhanced and have the descriptive ability to encode global context.

The spatial attention demonstrated in Figure 2.5 is a variant of the self-attention function designed for point cloud processing. Given the input feature matrix $F \in \mathbb{R}^{N \times C'}$ for a point set, F is projected to three different

feature subspaces to form the query, key and value which are FU, FV, $FT \in \mathbb{R}^{N \times C'}$ respectively. FU, FV and FT are calculated as the following:

$$FU = \alpha_s(F), FV = \beta_s(F), FT = \gamma_s(F) \tag{2.6}$$

where α_s, β_s and γ_s are the transformation functions achieved through different fully connected (FC) layers. As the output features in the attention function in Vaswani et al. (2017) are computed through attending to all positions, a spatial attention matrix $SA \in \mathbb{R}^{N \times N}$ should be calculated to capture the relationships between all possible point pairs. Following the Fu et al. (2018), we use the dot product between the FV_j and the transpose of FU_i to represent the correlation between the point i and j in a point cloud:

$$sa_{ij} = softmax_j(FV_j \cdot FU_i^T) \tag{2.7}$$

where sa_{ij} is the normalized spatial attention map that estimates the impact of point j on point i. Similar features of two points give rise to a high correlation between them, contributing to a large value in sa_{ij}. The final feature F_{sa} is computed as the following:

$$F_{sa} = \alpha_{sa}\, SA \cdot FT + F \tag{2.8}$$

where a matrix multiplication is performed between the spatial attention SA and the transformed embeddings FT. The output is multiplied by α_{sa}, a learnable scale parameter, and then element-wisely summed with the input feature F. The resulting feature F_{sa} encodes the embeddings across all point positions and this global view helps similar semantic features to achieve mutual gains, therefore improving the semantic consistency.

In addition to spatial attention, channel attention is employed to exploit channel-wise interdependencies. Every channel in high level features can be taken as a class-specific response and responses of different semantics are related to each other. Therefore, modelling the interdependencies between different channels can improve feature discriminability.

The structure of the channel attention model is shown in Figure 2.5. $CA \in \mathbb{R}^{C' \times C'}$ is the attention matrix directly computed from the matrix multiplication between the transpose of the input feature F and F. ca_{ij} measures the influence of j^{th} channel on i^{th} channel and CA estimates the dependency between all channels. Similar to the calculation of the spatial attention module, the resulting feature F_{ca} is computed as the following:

$$ca_{ij} = softmax_j(F_i^T \cdot F_j) \tag{2.9}$$

$$F_{ca} = \beta_{ca}\, F \cdot CA + F \tag{2.10}$$

Where a matrix multiplication is performed between CA and F. The output is multiplied by a learnable scale parameter β_{ca} and then element-wisely summed with the input feature F.

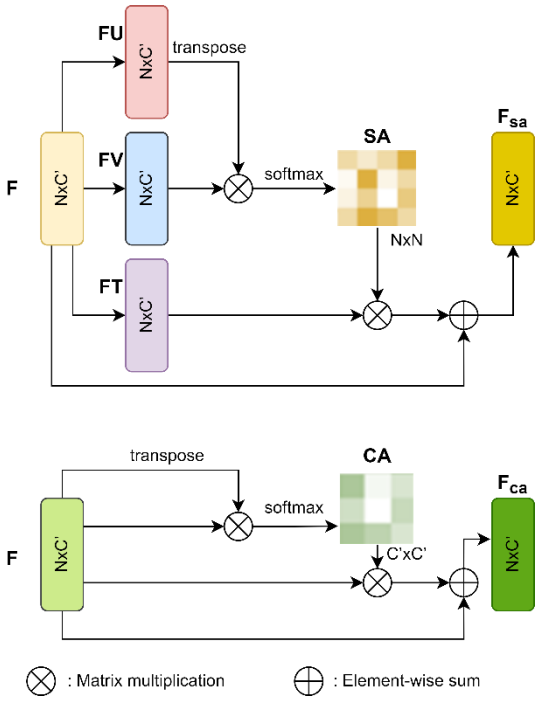


Figure 2.5 Structures of spatial attention (top) and channel attention (bottom).

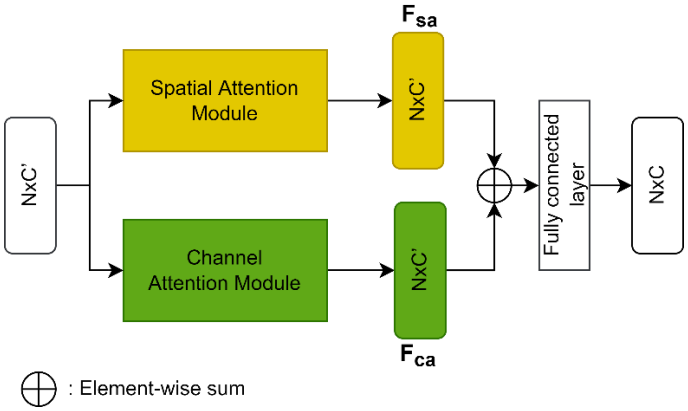


Figure 2.6 Structure of spatial-channel attention model.

In order to make full use of the global context, F_{sa} and F_{ca} are summed up and the sum is passed to an FC layer to obtain pointwise semantic labels. The structure is shown in Figure 2.6, where C is the number of classes for the final prediction. With this spatial-channel attention module, pointwise features are updated from a global perspective. The complicated interactions between points are comprehensively learned, contributing to more accurate predictions.

Attention based modules are also used in RandLA-Net (Hu et al., 2020) which achieves state-of-the-art in many semantic segmentation of point clouds tasks. Although RandLA-Net and our method both apply attention functions to improve the network performance, our spatial-channel attention is different from the attentive pooling in RandLA-Net in the following perspectives. First, RandLA-Net searches KNN neighbours and aggregates their features to describe the local geometry, while our spatial attention pays attention to all points in the input cloud. Even though RandLA-Net can progressively enlarge the receptive field through step-by-step subsampling, detailed geometrical information is lost as a result of subsampling. In contrast, the spatial attention avoids this issue by considering all the points from a global view. Second, the attention scores in RandLA-Net are calculated by putting a feature vector to a learnable MLP layer followed by a softmax function. In comparison, the attention scores in our method are based on the correlation between different points for spatial attention and the interdependencies between different channels for channel attention. Third, in RandLA-Net, the attention scores can be taken as a mask to select important features of each neighbour. Then feature vectors of K nearest neighbours are summed up into one informative feature vector to capture the local geometry for the central point. In our spatial-channel attention model, the spatial attention module searches for the important position from the entire input point clouds and aggregates information from all other points by a weighted sum. The channel attention module reveals the interdependencies between different channels, and the output features gather all useful information from all other channels. Considering representative features are already extracted at point and object levels by Hybrid KPConv and SegECC, the attentive pooling proposed in RandLA-Net to extract local geometrical features is not necessary for our network and therefore, we apply the spatial-channel attention to optimize network prediction from a global perspective.

2.3.4 Overall Network architecture

With three blocks introduced above, LGENet that encodes both local and global information can be constructed for semantic segmentation of ALS point clouds. Following the fully convolutional network proposed

by Thomas et al. (2019), our network is composed of an encoder and a decoder. As illustrated in Figure 2.7, the encoder has 5 convolutional layers and each layer consists of two convolutional blocks. We use hybrid KPConv for all convolutional blocks. However, SegECC is only inserted at the second block of the third and fourth layer. According to Feng et al. (2020), using blocks that encode global features in all layers fails to improve model performance because those blocks intensively increase the number of network parameters and it is difficult to achieve a global optima. This is also demonstrated by our experiments shown in Table 2.5. In order to capture the local geometry at multiple scales, downsampling is used to enlarge the receptive field of convolutions step by step.

In the decoder, nearest upsampling is employed to obtain final pointwise features. Four skip connections are applied to pass intermediate features from encoder to decoder. Those features are concatenated with the upsampled features and then passed to a unary block which is a 1×1 convolution. At end the of the network, a spatial-channel attention block is stacked to consider global context, thus improving final pointwise semantic predictions.

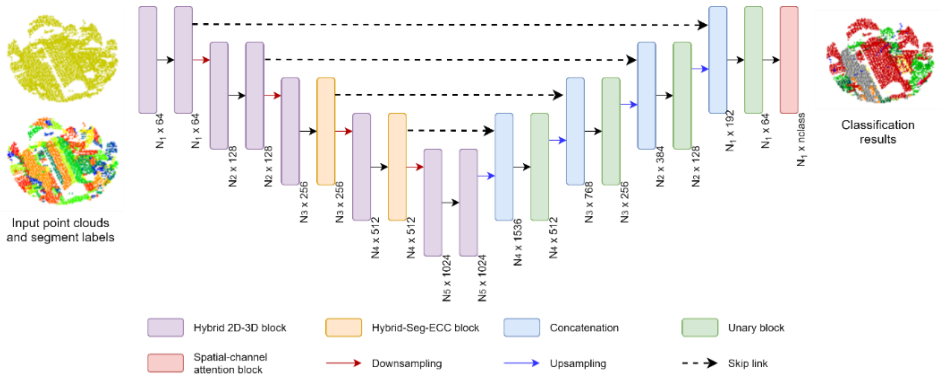


Figure 2.7 Illustration of the proposed LGENet architecture for semantic segmentation of ALS point clouds. The encoder consists of hybrid 2D-3D blocks and hybrid-SegECC blocks. The decoder is composed of unary blocks (1×1 convolutions). N1 > N2 > N3 > N4 > N5 denote point numbers. Intermediate features are passed from encoder to decoder through four skip links. The spatial-channel attention block is stacked at the end of the network.

For network training, we use the weighted cross entropy loss shown in (2.12). The weight for each class w_c is calculated by its proportion out of the total number of points, shown in (2.11), where N_c denotes the number of points for cth class. In (2.12), $\hat{y}_{ic}$ represents whether the

ground truth label for ith is in cth category and ρ_{ic} represents the corresponding predicted probability.

$$w_c = \frac{1/\gamma_c}{\sum_{c=1}^{C}\frac{1}{\gamma_c}} \quad , \quad \gamma_c = \frac{N_c}{\sum_{c=1}^{C} N_c} \tag{2.11}$$

$$L_{weighted} = -\frac{1}{N_{in}}\sum_{i=1}^{N_{in}}\sum_{c=1}^{C} w_c \hat{y}_{ic}\, log\rho_{ic} \tag{2.12}$$

2.4 Experiments

In this section, experiments are shown to evaluate the effectiveness of the proposed network in two ALS datasets. We compare the performance of our model against that of other the state-of-the-art models on the ISPRS benchmark (Niemeyer et al., 2014). We also conduct a comprehensive experiments on the ISPRS benchmark (Niemeyer et al., 2014) to show the effectiveness of our proposed method and evaluate how hyperparameters and network structure influence the model performance. Next the DFC2019 dataset (Bosch et al., 2019) is used to further demonstrate the advantages of our method.

2.4.1 Experiments on ISPRS benchmark dataset

2.4.1.1 Dataset

The performance of our method is evaluated by the ISPRS benchmark dataset of 3D labelling (Niemeyer et al., 2014). An overview of the dataset is shown in Figure 2.8. The benchmark dataset was obtained in August 2008, at Vaihingen, Germany, through a Leica ALS50 system whose mean flying height is 500m and field of view is 45°. Point clouds were captured with a density of 4 points/m^2. Each point has five attributes, namely, XYZ coordinates, intensity values and number of returns. The dataset is labelled into 9 classes, including powerline, low vegetation, impervious surface, car, fence/hedge, roof, façade, shrub, and tree. In ISPRS 3D labelling contest, the point cloud is divided into a training area and a testing area. The training area consists of 753,876 points, dominated by residential buildings. The testing area contains 411,722 points located in the city centre.

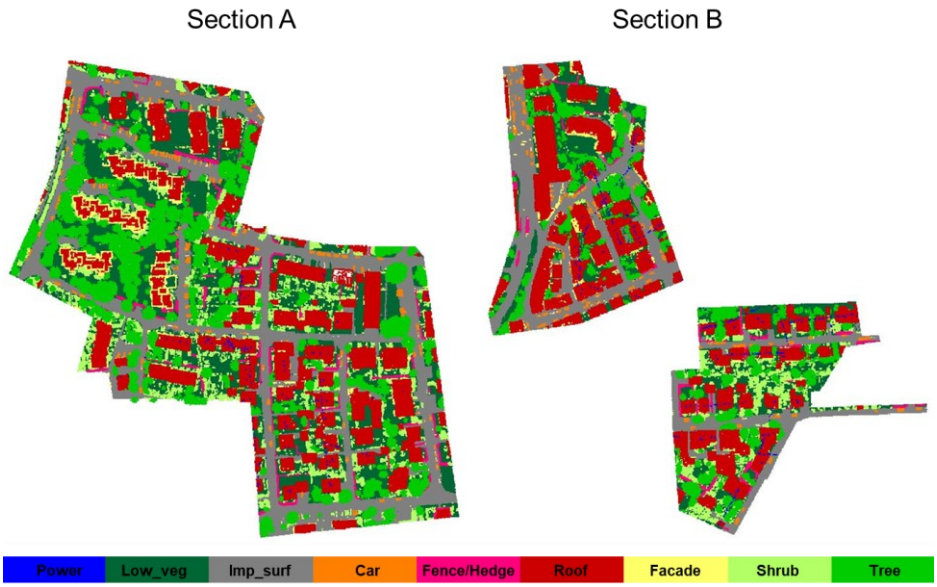


Figure 2.8 An overview of the ISPRS benchmark dataset. Section A is used for model training and Section B is used for model evaluation.

2.4.1.2 Accuracy assessment

Following the evaluation metrics of ISPRS benchmark dataset, we use the average F1 score (Avg. F1) and overall accuracy (OA) to evaluate our method. The overall accuracy measures the percentage of correctly predicted points in the total number of test points. F1 score is a statistical metric calculated from precision and recall.

$$precision = TP \,/\, (TP + FP) \tag{2.13}$$

$$recall = TP \,/\, (TP + FN) \tag{2.14}$$

$$F1\ score = 2 \times (precision \times recall) / (precision + recall) \tag{2.15}$$

Where TP, FN and FP are true positives, false negatives and false positives receptively in a confusion matrix.

2.4.1.3 Preprocessing

The ISPRS benchmark dataset is first segmented by the algorithm proposed by Landrieu and Obozinski (2017) to obtain segment labels required in the SegECC block. We use both XYZ coordinates and intensity as the input of the segmentation algorithm. The most important factor of the segmentation is the regulation strength, determining the coarseness of the final partition. The regulation strength is set to 0.03 in this chapter.

When preparing the data for training, we subsample the point clouds with a grid sampling size of 0.24m, in order to deal with the large variation in point density in ALS point clouds. Spheres are randomly selected from the subsampled point clouds and fed into the network. The radius of the input sphere is taken as 24m. We use intensity, absolute Z- coordinates and normalized Z- coordinates within the sphere as input features. For data augmentation, the input sphere is randomly rotated around the Z-axis to improve the network robustness to orientation. Also, random noises are added to XYZ coordinates with a σ of 4 cm which is chosen empirically and will not significantly modify the geometry of target objects.

2.4.1.4 Network implementation

As mentioned in Section 2.4.1.3, input point clouds are downsampled in different layers. Table 2.1 shows the grid size of the downsampling and the size of convolution kernels from layer 1 to layer 5. The convolution radius is 2.5 times the grid size in the corresponding layer. For example, the input of the first convolutional layer is subsampled with the grid size of 0.24m and the radius of the convolution in the first layer is 0.6m The number of kernel points in 3D KPConv is 15 and that of 2D KPConv is 17. Kernel points are initialized by the energy function proposed by Thomas et al. (2019) to ensure they are far from each other inside a given sphere (or circle for the 2D kernel).

Table 2.1 Subsampling grid size and convolution radius in differet layers.

Layer	1	2	3	4	5
Subsampling grid size (m)	0.24	0.48	0.96	1.92	3.84
Convolution radius (m)	0.6	1.2	2.4	4.8	9.6

2.4.1.5 Training and testing

The proposed network is implemented based on the PyTorch framework (Paszke et al., 2019), trained with a Geforce RTX 2080 Ti GPU. Stochastic gradient descent (SGD) optimiser is applied to optimize network weights. The weighted cross entropy loss function is applied to rebalance imbalanced data. During the training, we take 2000 iterations as one epoch. The learning rate starts from 0.001 with a decay rate of 0.9 at every 5 epochs. The model is trained for 60 epochs until the convergency is achieved. For testing, we randomly select spheres in the test area and each point is repeatedly fed into the network at least 20 times to obtain averages predictive probability. This repetition is to avoid the misclassification on points near the sphere boundary whose geometry may be incomplete.

2.4.1.6 Classification results

Qualitative results are shown in Figure 2.9 and the corresponding error map is demonstrated in Figure 2.10. It can be seen that the proposed LGENet correctly predicts most of the points in the testing area (Figure 2.10). As shown in Figure 2.9, car and façade points are well predicted, even though they have fewer instances in the whole dataset. Also, the LGENet can effectively identify powerline points although they are sparsely distributed above all other classes.

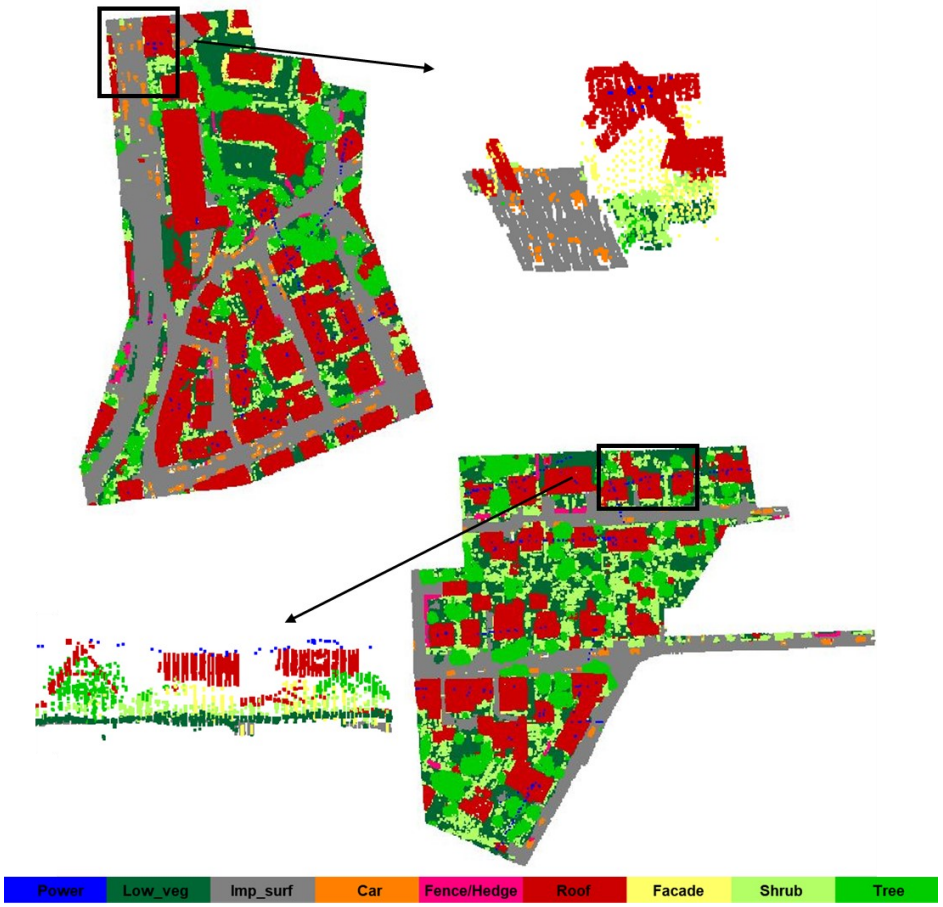


Figure 2.9 Classification results of our LGENet on the ISPRS benchmark dataset.

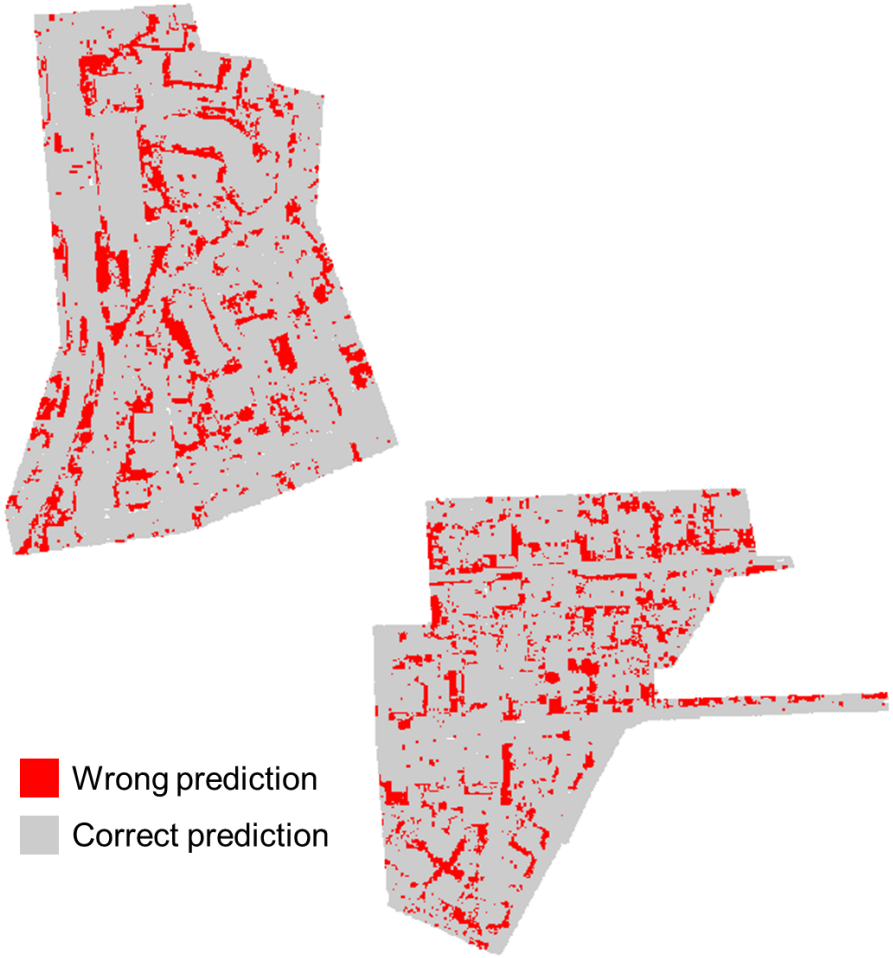


Figure 2.10 The error map of our LGENet on the ISPRS benchmark dataset.

Classification results are quantitively present by the confusion matrix in Table 2.2. LGENet can effectively recognize most of the classes with an overall accuracy of 0.845. Powerline, low vegetation, impervious surface, roof and tree points are well recognized. The worst classification result lies in the fence/hedge, most likely it will be predicted as shrub points according to the confusion matrix. This confusion is due to these classes being similar in geometry, pulse intensity and spatial distribution. In addition, façade points are also likely to be identified as shrub points since points are relatively sparse on façade in ALS dataset and they are difficult to separate if shrub points are very close to the building.

Table 2.2 Confusion matrix of our proposed network on ISPRS benchmark dataset. Precision, recall and F1 score are reported for each class. The overall accuracy is 0.845 and the average F1 score is 0.737.

	Power	Low_ veg	Imp_ surf	Car	Fence/ Hedge	Roof	Facade	Shrub	Tree
Power	459	2	0	0	0	95	16	1	27
Low_veg	0	83454	5870	61	263	1201	391	5008	2442
Imp_surf	0	9318	91972	44	18	301	35	296	2
Car	0	206	144	2612	84	112	7	518	25
Fence/Hedge	0	871	103	5	2063	188	33	3217	942
Roof	112	3883	114	3	60	101146	1486	1229	1015
Facade	14	776	77	34	34	1243	6583	1863	600
Shrub	1	4682	75	57	97	1281	368	14319	3938
Tree	14	1391	15	4	126	938	200	6144	45394
Precison	0.765	0.798	0.935	0.926	0.752	0.950	0.722	0.439	0.835
Recall	0.765	0.846	0.902	0.704	0.278	0.928	0.587	0.577	0.837
F1	0.765	0.821	0.918	0.800	0.406	0.938	0.647	0.499	0.836

2.4.1.7 Comparison to state-of-the-art methods

We quantitatively compare our LGENet to other state-of-the-art models on the ISPRS benchmark dataset in Table 2.3. LUH (Niemeyer et al., 2016) relies on handcrafted features and applies a two-layer hierarchical CRF at point and segment level. Other methods are based on deep learning, namely, WhuY4 (Yang et al., 2018), RIT_1 (Yousefhussien et al., 2018), alsNet (Winiwarter et al., 2019), A-XCRF (Arief et al., 2019), D-FCN (Wen et al., 2020), and Li et al. (2020).

Compared with all the above methods, LGENet achieves superior classification performances to other methods in terms of the average F1 score (0.737). Nevertheless, the overall accuracy of LGENet is slightly lower than the highest overall accuracy (0.850) obtained by A-XCRF. One explanation for this is that we apply a weighted loss to balance the imbalanced class distribution in the ISPRS benchmark dataset. The focusing on overall accuracy leads to bias on dominant categories and ignores minority classes in the dataset. Therefore, it is more meaningful to evaluate model performance by average F1 scores that equally reflect the model performance for all categories. Our LGENet significantly improves the baseline (KPConv) by 0.031 in the average F1 score and overall accuracy in 0.028. As the data processing

and hyper-parameter settings are the same for LGENet and the baseline network, the accuracy improvement is a result of our network design.

Table 2.3 Quantitative comparisons between our LGENet and other models on the ISPRS benchmark dataset. The F1 scores for different classes are shown in the first nine columns and the overall accuracy and the average F1 score are shown in the last two columns. The boldface text means the highest value in the column.

	Power	Low_ veg	Imp_ surf	Car	Fence/ Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
LUH (Niemeyer et al., 2016)	0.596	0.775	0.911	0.731	0.340	0.942	0.563	0.466	0.831	0.684	0.816
WhuY4 (Yang et al., 2018)	0.425	**0.827**	0.914	0.747	**0.537**	0.943	0.531	0.479	0.828	0.692	0.849
RIT_1 (Yousefhussien et al., 2018)	0.375	0.779	0.915	0.734	0.180	0.940	0.493	0.459	0.825	0.633	0.816
alsNet (Winiwarter et al., 2019)	0.701	0.805	0.902	0.457	0.076	0.931	0.473	0.347	0.745	0.604	0.806
A-XCRF (Arief et al., 2019)	0.630	0.826	**0.919**	0.749	0.399	**0.945**	0.593	0.507	0.827	0.711	**0.850**
D-FCN (Wen et al., 2020)	0.704	0.802	0.914	0.781	0.370	0.930	0.605	0.460	0.794	0.707	0.822
Li et al. (2020)	0.754	0.820	0.916	0.778	0.442	0.944	0.615	0.496	0.826	0.732	0.845
KPConv (Thomas et al., 2019)	0.735	0.787	0.880	0.794	0.330	0.942	0.613	0.457	0.820	0.706	0.817
Ours (LGENet)	**0.765**	0.821	0.918	**0.800**	0.406	0.938	**0.647**	**0.499**	**0.836**	**0.737**	0.845

2.4.1.8 Experiments with different model settings

2.4.1.8.1 Effectiveness of hybrid convolution

To justify the importance of 2D KPConv in semantic segmentation of ALS point clouds, we conduct experiments to compare and contrast models with and without 2D convolutions. We also evaluate how the model performance changes with a different number of kernel points in 2D convolutions. Furthermore, we test the model performance when only using 2D convolutions and when searching neighbours among projected 2D points in the 2D KPConv of hybrid blocks.

Table 2.4 presents the quantitative results using different convolutions. The original 3D KPConv is used as the baseline. Although deformable KPConv kernels (Thomas et al., 2019) are adaptive to object surface and enhance descriptive power of output features, they fail in the experiments on the ISPRS benchmark dataset (last row in Table 2.4). According to Thomas et al. (2019), this is because the dataset has only

9 classes and lacks object diversity comparing to other datasets with more complex scenes.

Table 2.4 Quantitative results (F1 scores) of hybrid KPConv with different numbers of kernel points in the 2D convolution on ISPRS benchmark dataset. Here, we fixed the number of kernel points in 3D convolution as 15. The baseline network uses rigid 3D KPConv proposed by Thomas et al. (2019). The hybrid models involve 2D KPConv in all convolutional layers in the network. The number in the bracket represents the number of kernel points in 2D KPConv. The fifth row shows the results of the model only using 2D KPConv. The sixth row shows the results of hybrid blocks searching neighbours among projected 2D points. The seventh row shows the results of the deformable KPConv.

	Power	Low_ veg	Imp_ surf	Car	Fence/ Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
Base	0.735	0.787	0.880	0.794	0.330	0.942	0.613	0.457	0.820	0.706	0.817
Hybrid (5)	0.657	0.806	0.909	0.756	0.365	0.938	0.627	0.486	0.807	0.706	0.829
Hybrid (9)	0.693	0.803	0.900	0.762	0.363	0.937	0.632	0.497	0.824	0.712	0.829
Hybrid (17)	0.703	0.811	0.908	0.757	0.381	0.939	0.632	0.495	0.826	**0.717**	**0.837**
Only 2D (17)	0.637	0.741	0.853	0.794	0.389	0.862	0.629	0.380	0.793	0.675	0.777
Hybrid (17) 2D neighbours	0.651	0.801	0.889	0.755	0.347	0.932	0.625	0.460	0.809	0.697	0.823
Deformable kernels	0.604	0.743	0.879	0.734	0.403	0.941	0.595	0.453	0.820	0.686	0.812

By comparing the baseline to the only-2D network, it can be seen that although the 2D convolutions lead to low F1 scores in most of the semantic classes, the only-2D network outperforms the baseline with 3D convolutions for the fence/hedge class which is a very difficult class for other methods. A possible explanation for this could be that fence/hedge are elongated structures distributed on XY plane and fixed kernel points on XY planes contributes to better representations. For rigid 3D convolutions, kernel points are distributed in the sphere and very limited kernel points are located near the XY plane, resulting in the failure on those elongated structures distributed on the ground. When combining 2D and 3D KPConvs, we could see better average F1 and overall accuracy are obtained with more kernel points in 2D KPConv (Hybrid(5), Hybrid(9) and Hybrid(17) in Table 2.4. Using 2D KPConv leads to more confusion between powerline points and roof points because projecting points to an XY plane is likely to cause the overlap between the powerline points and roof points and responses of 2D kernels for these two classes can be very similar, while this adverse impact is relieved when using more kernel points to enhance the

descriptive power of convolutions. The average F1 and overall accuracy achieved are 0.717 and 0.837 respectively when the hybrid convolution layers have 17 kernel points in the 2D convolution. Comparing to the baseline network, this combination significantly improves the F1 scores in fence/hedge and shrub by solving the confusion between them.

We also show the results of searching neighbours among projected 2D points in the 2D KPConv of hybrid blocks in the sixth row of Table 2.4. When comparing to the neighbour searching strategy mentioned in section 2.3.1 (Hybrid(17)), F1 scores for all categories are lower, especially powerline. This is probably because powerlines always hang over all other objects and searching neighbours among projected 2D points brings irrelevant points like impervious surface and façade points which only give noises and have no contribution to classification results.

2.4.1.8.2 Effectiveness of SegECC convolution

To take advantage of the SegECC operation, we place the SegECC at different hybrid convolutional layers in the network architecture and quantitative results are shown in Table 2.5. As shown in the first and second columns in Table 2.3, adding SegECC at 1 to 4 layers and 2 to 4 layers fails to improve the network performance in terms of average F1 and overall accuracy, compared with the network only using hybrid convolutional layers. This is in accordance with observations obtained by Feng et al. (2020) that adding more layers to encode global context in an encoder and decoder network deteriorates model performance. One possible explanation for this drop is that more SegECC blocks raise the number of network parameters and therefore the network fails to achieve global optima. When only inserting the SegECC at the fourth layer, F1 scores on most of the classes are quite similar to the results of the hybrid network, like roof and impervious surface points. The model performance achieves the best by adding the SegECC at layer 3 and layer 4. It outperforms the hybrid network (No SegECC in Table 2.5) in terms of average F1 score and overall accuracy by 0.007 and 0.006. The most significant increase lies in the powerline, façade and fence/hedge which only takes a small proportion of the training data and are difficult to predict. This suggests that the global contextual information captured at an object level is valuable for these classes. This is probably because distributions of these objects are unique in urban scenes. Façades exist with roofs. Powerlines are always above all other objects and fences/hedges always surround buildings. However, this global context fails to solve the confusion between shrub and tree because shrub and tree objects are always mixed distributed in urban scenes. Thus, exploiting global context at the object level is limited in distinguishing these two classes.

Table 2.5 Quantitative comparison of classification results using SegECC operations at different hybrid convolutional layers on ISPRS benchmark dataset. The last row shows the results of the network only use hybrid convolutional layers without SegECC operation.

	Power	Low_ veg	Imp_ surf	Car	Fence/ Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
1,2,3,4	0.651	0.806	0.902	0.704	0.310	0.936	0.631	0.417	0.795	0.683	0.821
2,3,4	0.725	0.815	0.911	0.750	0.377	0.925	0.615	0.432	0.797	0.705	0.829
3,4	0.740	0.819	0.916	0.749	0.420	0.941	0.649	0.466	0.814	**0.724**	**0.843**
4	0.773	0.808	0.915	0.745	0.381	0.936	0.642	0.446	0.815	0.718	0.834
No SegECC	0.703	0.811	0.908	0.757	0.381	0.939	0.632	0.495	0.826	0.717	0.837

According to Landrieu and Obozinski (2017), the regularization factor determines the number of segments produced by the algorithm. To evaluate how our method sensitive to this factor, we conduct experiments on the ISPRS benchmark dataset using segmentation results obtained from three different values of the regularization factor for the segmentation algorithm. Quantitative results are listed in Table 2.7 and qualitative results are demonstrated in Figure 2.11. Three values are tested in our experiments, namely 0.01, 0.03 and 0.1. The numbers of segments they yield in training and testing areas are shown in Table 2.6. A larger regularization factor leads to a smaller number of segments. In Figure 2.11, coarse segmentation results are obtained with a large regularization factor (0.1). This undersegmentation cannot separate delicate structures like fence/hedge from other objects and therefore prevents the network from learning interactions among different objects, resulting in poor semantic segmentation results shown in the last row in Table 2.7. Smaller regularization factors (0.01 and 0.03) produce more detailed segmentation and this allows the network to capture the interactions among different segments within a single object and relationships between different objects, thus, improving semantic segmentation results. Comparing the results of two small regularization factors, 0.03 gives better results than 0.01 in terms of the overall accuracy and it achieves better accuracy in low vegetation, impervious surface, roof and façade. These classes are large objects and segmentation results of 0.01 are too fragmented to assist the network to learn better representations. Therefore, we use 0.03 as the regularization factor to obtain segmentation labels before the network training.

Table 2.6 The number of segments produced with different regularization factors for the segmentation algorithm.

Reg. factor	0.01	0.03	0.1
Training area	46023	18756	2737
Test area	27774	10906	1583

Table 2.7 Comparison of model performance on ISPRS benchmark dataset when using different regularization factors in the segmentation algorithm.

Reg. factor	Power	Low_ veg	Imp_ surf	Car	Fence /Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
0.01	0.717	0.811	0.910	0.778	0.397	0.937	0.647	0.476	0.818	0.721	0.835
0.03	0.740	0.819	0.916	0.749	0.420	0.941	0.649	0.466	0.814	**0.724**	**0.843**
0.10	0.619	0.767	0.843	0.663	0.209	0.929	0.625	0.417	0.797	0.652	0.795

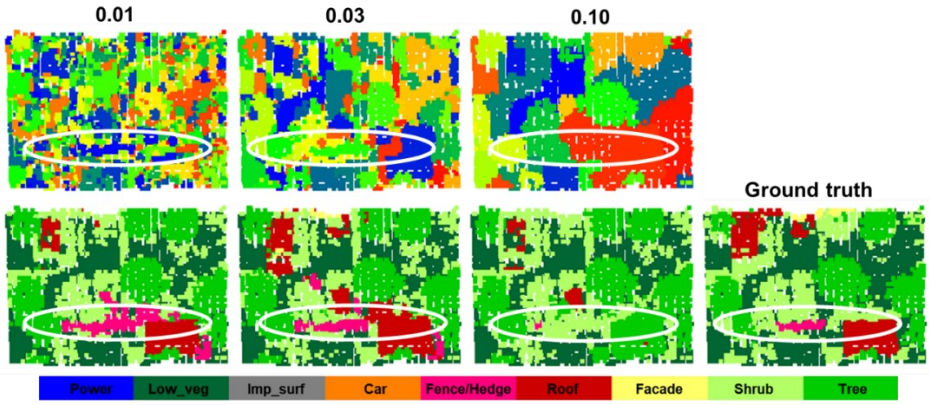


Figure 2.11 Qualitative classification results on ISPRS benchmark dataset with different regularization factors. The top row shows the segmentation results obtained from the unsupervised segmentation algorithm and different colours represent different segments. The bottom row presents the corresponding semantic segmentation results and ground truth.

We also experiment our SegECC with the segmentation results obtained from the algorithm proposed by Vosselman et al. (2017). Some examples of segmentation results are qualitatively shown in Figure 2.12 and network predictions on the ISPRS benchmark dataset are quantitively shown in Table 2.8. Segmentation results generated by the algorithm of Vosselman et al. (2017) fail to improve the network performance. The accuracies on car and fence/hedge are much lower, compared to the results obtained using clustering results of Landrieu and Obozinski (2017). This can be explained by the under-segmentation demonstrated in Figure 2.12. It can be seen that clustering results obtained by Vosselman et al. (2017) have more well-segmented planar components and are coarser than the output of Landrieu and Obozinski (2017). Car points are not separated from nearby tree points and fence/hedge points are grouped with nearby shrub points to form a single segment. Introducing the wrong

clustering information during training consequently deteriorates the network performance on those classes.

Table 2.8 Quantitative comparison of classification results using different segmentation methods for SegECC operation on ISPRS benchmark dataset. Seg1 uses the segmentation results obtained by Vosselman et al. (2017) and seg2 takes L_0-cut proposed by Landrieu and Obozinski (2017). The last row shows the results of the network only use hybrid convolutional layers without SegECC operation.

	Power	Low_veg	Imp_surf	Car	Fence/Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
seg1	0.675	0.797	0.858	0.754	0.324	0.926	0.634	0.436	0.806	0.690	0.814
seg2	0.740	0.819	0.916	0.749	0.420	0.941	0.649	0.466	0.814	**0.724**	**0.843**
no_seg	0.703	0.811	0.908	0.757	0.381	0.939	0.632	0.495	0.826	0.717	0.837

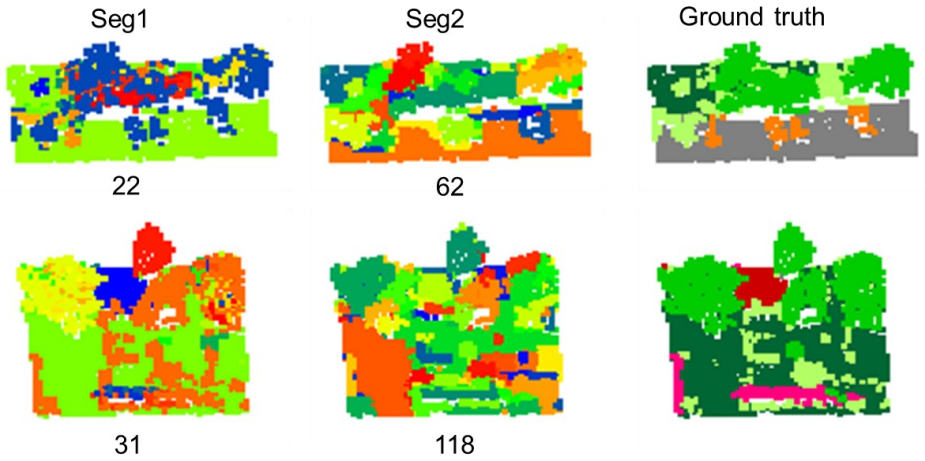


Figure 2.12 Examples of segmentation results on ISPRS benchmark dataset. Seg1 uses the segmentation results obtained by Vosselman et al. (2017) and seg2 takes L_0-cut proposed by Landrieu and Obozinski (2017). Different colours represent different segments. Numbers below the segmentation result are numbers of segments within the cropped point clouds.

In order to improve the model robustness and save GPU memory, we randomly select edges instead of using all edges during the training. Table 2.9 presents the classification results using a different number of edges in the SegECC operation. For simplicity, we select the same amount of edges in SegECC regardless of the layer. Considering the GPU memory and neighbourhood sizes in layer 3 and layer 4 (Figure 2.13), we try 3 values for the number of selected edges, namely 40,

80 and 120. It can be seen from Table 2.9 that the use of too many edges cannot improve model performance due to overfitting while using only a few edges is insufficient to exploit contextual information. Therefore, we randomly select 80 edges for SegECC in layer 3 and layer 4.

Table 2.9 Comparison of model performance on ISPRS benchmark dataset with a different number of edges selected in SegECC operation.

	Power	Low_veg	Imp_surf	Car	Fence/Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
40	0.707	0.809	0.905	0.727	0.402	0.939	0.643	0.448	0.814	0.711	0.831
80	0.740	0.819	0.916	0.749	0.420	0.941	0.649	0.466	0.814	**0.724**	**0.843**
120	0.696	0.801	0.903	0.760	0.359	0.930	0.625	0.437	0.801	0.701	0.822

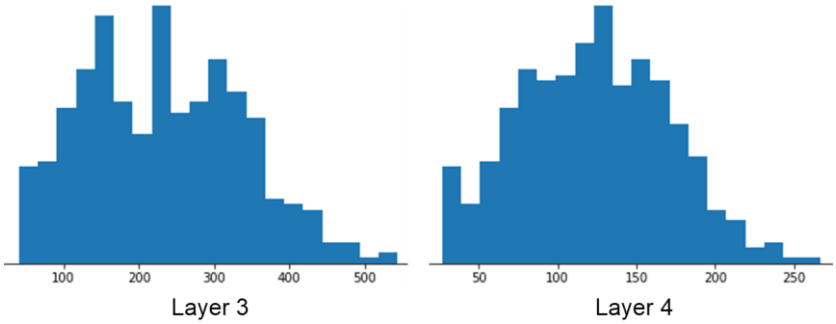


Figure 2.13 Distribution of neighbourhood size for Vaihingen dataset.

2.4.1.8.3 Effectiveness spatial-channel attention

Quantitative results of the model using spatial-channel attention on ISPRS benchmark dataset are shown in Table 2.10. Using spatial-channel attention does not significantly improve the overall accuracy but increases the average F1 score from 0.724 to 0.737. The F1 score in powerline, car and shrub increase by 0.025, 0.051 and 0.033 respectively. Figure 2.14 shows that with the spatial-channel attention, powerline and car points can be corrected by taking global contextual information. Also, the spatial-channel eliminates 'salt and pepper' effects in the classification results. In the bottom row of Figure 2.14, it removes isolated façade points to make results more consistent with surrounding points, although the model with the spatial-channel attention wrongly predicts hedge points to be shrub and tree points. These incorrect predictions lead to the slight decrease in the F1 score for fence/hedge shown in Table 2.10.

Table 2.10 Comparison of model performance with spatial-channel attention and without spatial-channel attention.

	Power	Low _veg	Imp_ surf	Car	Fence/ Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
Without spatial-channel attention	0.740	0.819	0.916	0.749	0.420	0.941	0.649	0.466	0.814	0.724	0.843
With spatial-channel attention (LGENet)	0.765	0.821	0.918	0.800	0.406	0.938	0.647	0.499	0.836	**0.737**	**0.845**

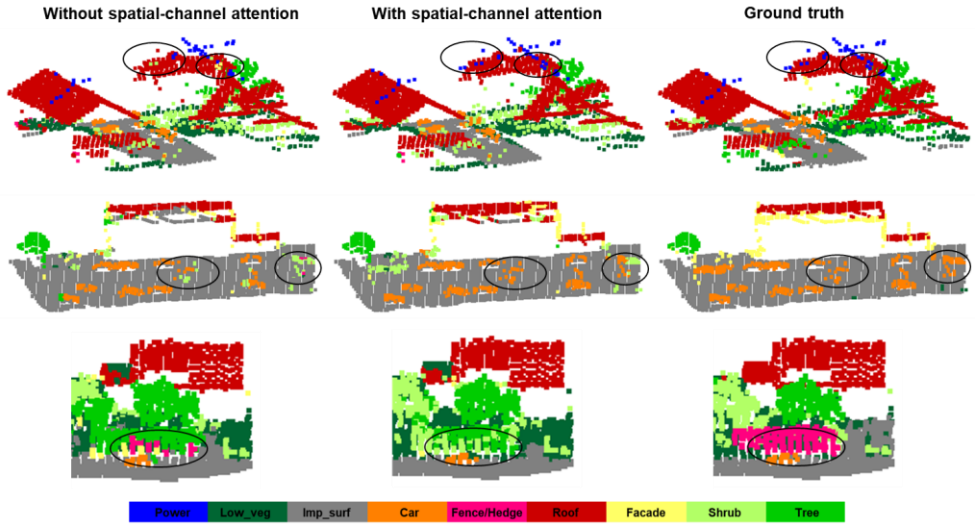


Figure 2.14 Qualitative comparison of model performance with spatial-channel attention and without spatial-channel attention.

2.4.1.9 Experiments with PointNet++ backbone

In this chapter, we mainly focus on adapting the KPConv network, while the proposed SegECC layer and spatial-channel attention (DA) is also possible to work with other feature extractors. In the following experiments, we insert SegECC and spatial-channel attention to PointNet++ (Qi et al., 2017b) and then test models on the ISPRS benchmark dataset. Table 2.11 shows the results of PointNet++, PointNet++ with a SegECC layer and PointNet++ with a SegECC layer and a spatial-channel attention. The model with both the SegECC layer and the spatial-channel attention achieves the best performance in terms of the average F1 score and the overall accuracy.

Table 2.11 Quantitative comparison of classification results of PointNet++, PointNet++ with a SegECC layer and PointNet++ with a SegECC layer and a spatial-channel attention on the ISPRS benchmark dataset.

	Power	Low_veg	Imp_surf	Car	Fence/Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
Pointnet++	0.604	0.814	0.904	0.723	0.103	0.906	0.349	0.469	0.739	0.623	0.806
Pointnet++ w/ SegECC	0.644	0.797	0.897	0.660	0.185	0.923	0.621	0.359	0.776	0.651	0.812
Pointnet++ w/ SegECC+DA	0.710	0.823	0.915	0.780	0.215	0.921	0.590	0.480	0.731	**0.685**	**0.826**

2.4.2 Experiments on DFC2019 dataset

2.4.2.1 Dataset

We also evaluate our LGENet by another ALS dataset published by the IEEE Geoscience and Remote Sensing Society (GRSS) for the Data Fusion Contest in 2019 (DFC2019) (Bosch et al., 2019). The DFC2019 dataset covers large-scale urban areas, about 100 km^2, in two large cities, namely, Omaha, Nebraska and Jacksonville, Florida in the Unites States. The ALS point clouds are captured with an aggregate nominal pulse spacing of 0.8m and contain about 200 million points in total. For each point, not only XYZ coordinates but also the intensity and return number are available. The point clouds are manually labelled into five classes, namely ground, high vegetation, building, water and bridge deck. We use 100 tiles as the training data and 10 tiles as the test dataset, same as the data split in Wen et al. (2020). We use the same hyperparameters as those for experiments on ISPRS benchmark dataset, except the radius of the input sphere and the grid cell size for subsampling which are set as 48m and 0.48m respectively. This is because DFC2019 point clouds are sparse and only large objects are required to be predicted like building and bridge deck. Furthermore, the model is trained for 150 epochs to achieve convergence.

2.4.2.2 Classification results

Table 2.12 Quantitative classification results of different models on the DFC2019 dataset. The first five columns show F1 scores for five classes. The last two columns list the average F1 score and OA respectively. The boldface text demonstrates the highest value in the column.

	Ground	High Vege-tation	Build-ing	Water	Bridge Deck	Avg. F1	OA
Baseline (KPConv)	0.991	0.975	0.893	0.434	0.694	0.797	0.978
Hybrid	0.992	0.974	0.925	0.444	0.735	0.814	0.980
Hybrid-SegECC	0.993	0.979	0.924	0.447	0.791	0.827	0.983
Hybrid-SegECC-DA (LGENet)	0.993	0.983	0.928	0.474	0.791	**0.834**	**0.984**

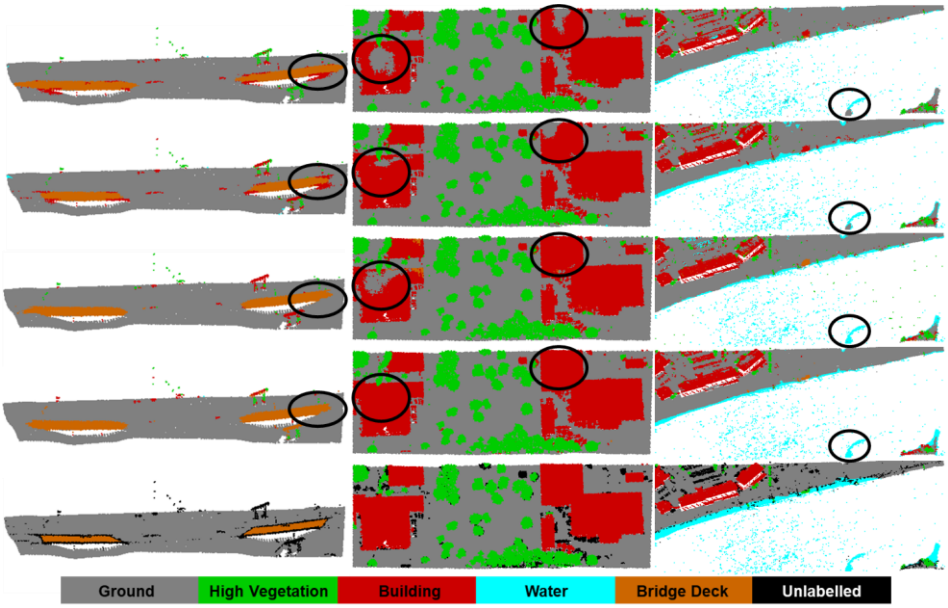


Figure 2.15 Some examples of classification results on the DFC2019 dataset obtained from different models. First row: baseline (KPConv), second row: hybrid convolution, third row: hybrid convolution with SegECC operations, fourth row: hybrid convolution with SegECC operations and spatial-channel attention at the end of the network, fifth row: ground truth.

Table 2.13 Quantitative comparisons between other methods and our LGENet on the DFC2019 dataset. The first five columns show F1 scores for five classes. The last two columns list average F1 scores and OA. The boldface text means the highest value in the column.

	Ground	High Vegetation	Building	Water	Bridge Deck	Avg. F1	OA
PointNet++ (Qi et al., 2017b)	0.983	0.958	0.797	0.044	0.073	0.571	0.927
PointSIFT (Jiang et al., 2018)	0.986	0.970	0.855	0.464	0.604	0.776	0.940
PointCNN (Li et al., 2018)	0.987	0.972	0.849	0.441	0.653	0.780	0.938
D-FCN (Wen et al., 2020)	0.991	0.981	0.899	0.450	0.730	0.810	0.956
DANCE-NET (Li et al., 2020)	0.991	0.939	0.870	**0.583**	0.839	**0.844**	0.968
GACNN (Wen et al., 2021)	**0.993**	0.968	0.911	0.425	**0.844**	0.828	0.951
Ours (LGENet)	**0.993**	**0.983**	**0.928**	0.474	0.791	0.834	**0.984**

Table 2.12 and Figure 2.15 present the classification results of our method on the DFC2019 dataset. The use of hybrid convolutions only gives rise to the F1 score increase in building, water and bridge deck by 0.032, 0.009 and 0.041 respectively compared to the baseline KPConv. The average F1 score is therefore increased by 0.017. When adding SegECC to the network (third row in Table 2.12), it can be seen that the F1 score of the bridge deck rises by 0.056. The contextual information at the segmentation level helps to correct the building points near the bridge deck (Figure 2.15). The effectiveness of the spatial-channel attention at the end of the network can be noticed by comparing the third and fourth rows in Table 2.12. Pointwise predictions are further corrected by considering contextual information from a global perspective. In order to demonstrate the advantages of our LGENet, we also compare our results on DFC2019 with others' results, namely PointNet++ (Qi et al., 2017b) PointSIFT (Jiang et al., 2018) PointCNN (Li et al., 2018), D-FCN (Wen et al., 2020), DANCE-NET (Li et al., 2020) and GACNN (Wen et al., 2021) shown in Table 2.13. Our LGENet achieves the best F1 scores on ground, high vegetation and building and it also reaches the top overall accuracy.

2.5 Conclusion

We proposed a novel network LGENet for semantic segmentation of ALS datasets. The LGENet learns representative features from local to global and exploits contextual information at both object and point levels. We first add 2D point convolutions to the 3D point convolutions of KPConv forming a hybrid block in order to enhance the learning of the representative local geometry, especially for elongated objects distributed on the horizontal plane. Next, segment-based edge conditioned convolution (SegECC) is inserted at the end of the hybrid block to encode the context at the object level. Segment labels used in SegECC are obtained before the training from an unsupervised segmentation algorithm, while the segment features are dynamically calculated from changing pointwise features during the training. We finally add a spatial-channel attention module to further improve the semantic predictions by considering global relationships between points and interdependencies between channels.

We verify the advantages of our proposed method by comprehensive experiments on the ISPRS benchmark dataset. LGENet outperforms the baseline model, KPConv (Thomas et al., 2019), by 0.031 in overall accuracy and 0.028 in average F1 score. When comparing to other state-of-the-art models, LGENet achieves the best accuracy in terms of the average F1 score (0.737). The overall accuracy is comparable with the published results obtained from those current leading models. Furthermore, we conduct experiments on the DFC2019 dataset and LGENet produces more accurate pointwise semantic predictions in terms of overall accuracy (0.984) and average F1 score (0.834), compared to the baseline and other leading models. This further demonstrates the advantage of our proposed LGENet.

Chapter 3 – Active and Incremental Learning for Semantic Airborne Laser Scanning Point Cloud Segmentation [2]

[2] This chapter is based on:

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2020. Active and incremental learning for semantic ALS point cloud segmentation. ISPRS Journal of Photogrammetry and Remote Sensing 169, 73–92.

Abstract

Supervised training of a deep neural network for semantic segmentation of point clouds requires a large amount of labelled data. Nowadays, it is easy to acquire a huge number of points with high density in large-scale areas using current LiDAR and photogrammetric techniques. However, it is extremely time-consuming to manually label point clouds for model training. In this chapter, we propose an active and incremental learning strategy to iteratively query informative point cloud data for manual annotation and the model is continuously trained to adapt to the newly labelled samples in each iteration. We evaluate the data informativeness step by step and effectively and incrementally enrich the model knowledge. The data informativeness is estimated by two data dependent uncertainty metrics (point entropy and segment entropy) and one model dependent metric (mutual information). The proposed methods are tested on two datasets. The results indicate the proposed uncertainty metrics can enrich current model knowledge by selecting informative samples, such as considering points with difficult class labels and choosing target objects with various geometries in the labelled training pool. Compared to random selection, our metrics provide valuable information to significantly reduce the labelled training samples. In contrast with training from scratch, the incremental fine-tuning strategy significantly save the training time.

3.1 Introduction

Point clouds, collections of points in 3D space, are characterized by their powerful abilities to represent position, size, shape and orientation of objects. Interpretation of point clouds captured by airborne laser scanning (ALS) systems is an essential step for many applications, such as 3D city modelling and urban land administration. Manually identifying urban objects like buildings, trees, and bridges requires a huge amount of human effort. To reduce this time-consuming and tedious work, researchers put their efforts on investigating the potential of machine learning techniques to deal with point cloud understanding (e.g. semantic segmentation) automatically.

Supervised machine learning is the most commonly used technique in point cloud interpretation. It relies on labelled data to train statistical models. A lot of models have been researched for the task of semantic point cloud segmentation, like Random Forests (RF) (Chehata et al., 2009), Supported Vector Machine (SVM) (Lodha et al., 2006), Gaussian Mixture Model (GMM) (Weinmann et al., 2014), AdaBoost (Lodha et al., 2007) and Artificial Neural Networks (ANN) (Xu et al., 2014). Recently, deep neural networks made significant breakthroughs in point cloud classification and segmentation tasks, such as PointNet (Qi et al., 2017a) and PointCNN (Li et al., 2018). Although the deep learning paradigm shows its power in complicated feature representation, it requires a massive amount of ground truth data to avoid overfitting during the training. Unfortunately, the ground truth for the semantic point cloud segmentation requires pointwise labelling, which is very time-consuming when done manually. To label ALS point clouds covering 2 km^2 in Dublin centre into 8 categories, over 2500 hours were spent with an appropriate tutorial, supervision, and careful cross-checks to minimize the error (Zolanvari et al., 2019). Therefore, strategies should be proposed to alleviate this manual annotation effort.

To reduce manual annotation efforts, one possible strategy is to bring labels from other data sources. For example, Yang et al. (2020) directly bring 2D labels from topographic maps to train the model for semantic segmentation of point clouds. Another solution is to effectively train models with only a small set of labelled data. Semi-supervised learning is one of the techniques proposed to train models with limited labelled data and a large amount of unlabelled data (Zhu and Goldberg, 2009). It takes advantages of unlabelled data in order to facilitate supervised learning tasks in the lack of labelled data.

Identifying and only labelling the most informative samples is another promising alternative. Settles (2009) show that the informativeness of

training samples differs. Some samples are informative and therefore improve the model performance, while some bring less information and others are even outliers to models. Only a part of the annotated data determines the models' parameters. Therefore, an efficient learning strategy should be investigated to select the most informative samples for model optimization. Then manual annotation can be reduced because only selected informative samples need to be manually annotated. Active learning is an efficient learning strategy proposed to solve the problem.

The aim of active learning is to create a small training subset from a larger unlabelled dataset. The strategy is to assess the sample informativeness in the unlabelled pool using the current model state. Then informative samples are manually annotated and added to the current training data for the next training. This minimizes manually labelling efforts during ground truth preparation while keeping the model performance in a supervised learning process. Active learning has been studied in many tasks like natural language processing (Wang et al., 2019a), object detection (Kellenberger et al., 2019), image classification (Wang et al., 2017), image semantic segmentation (Vezhnevets et al., 2012) and remote sensing (Tuia et al., 2011). Nevertheless, how to perform point cloud labelling tasks by active learning is rarely researched (Feng et al., 2019; Lin et al., 2020b; Luo et al., 2018).

In addition to the active learning that iteratively selects informative samples for training, incremental learning is a type of machine learning technique where the learning process occurs whenever new data emerge and the current learned knowledge is adjusted according to the new data. In incremental learning, the model knowledge is continuously enlarged by the continually added samples. Incremental learning has been investigated in many computer vision applications like object recognition (Bai et al., 2015), image classification (Ristin et al., 2016) and segmentation (Tasar et al., 2018), visual tracking (Dou et al., 2015) and surveillance (Shin et al., 2018). Some researches involve active selection processes in incremental learning for image related tasks (Brust et al., 2020; Zhou et al., 2017). However, to our knowledge, there is no research to integrate active learning with incremental learning for semantic segmentation of ALS point clouds.

In this chapter, we propose an effective framework for semantic segmentation of large-scale ALS point clouds in urban areas based on both active and incremental learning techniques. The objective of this chapter is to effectively query informative samples that can improve the performance of deep learning models meanwhile minimizing the

annotation efforts required for training data preparation. Also, we alleviate the training efforts by implementing incremental learning. We assess the informativeness of point clouds by three uncertainty metrics, point entropy, segment entropy and mutual information. The major **contributions** of this chapter are as follows:

1) We introduce an active and incremental learning framework to effectively reduce the number of training samples required by deep neural networks for semantic segmentation of large ALS point clouds.
2) To identify the most informative parts of a point cloud for model training, we quantitatively assess both data dependent and model dependent uncertainties using three independent metrics. The data dependent uncertainty is estimated by point entropy and segment entropy. The segment entropy considers interactions among neighbouring points. Model dependent uncertainty is estimated by mutual information which analyses the disagreements produced by different model parameters.
3) To make use of the knowledge obtained from previous training and reduce training efforts in the task of semantic segmentation of large ALS point clouds, we allow the model to incrementally learn from the point clouds by fine-tuning the model obtained from the previous stage instead of training the model from scratch for each active learning iteration.

The rest of the chapter is structured as the following. Section 3.2 reviews related work. Section 3.3 introduces the proposed active and incremental learning framework and describes the network structure as well as three query functions. Experimental results of two subsets of the AHN3 dataset are presented and analysed in Section 3.4. Section 3.5 concludes the main observations in the chapter and provides some suggestions for the future work.

3.2 Related work

3.2.1 Deep learning approaches

Recently, deep learning algorithms have been improving accuracy on semantic segmentation of point clouds. There are two groups of methods, 2D and 3D. For 2D methods, point clouds are firstly projected onto 2D image planes and then passes into Convolutional Neural Networks (CNNs). For example, Kalogerakis et al. (2017) and Boulch et al. (2018) capture point cloud images from different views and take these images as the input of image based CNNs. The output pixelwise

image labels are then projected back to 3D points. However, the work of Kalogerakis et al. (2017) only deals with part segmentation of objects instead of assigning pointwise labels for either indoor or outdoor scenes which are much more complex. Although SnapNet (Boulch et al., 2018) is designed for semantic segmentation of point clouds in complex scenes, during the projection, self-occlusion is inevitable, especially for complicated scenes. To deal with ALS data covering large areas, some researchers project point clouds onto 2D grids from the top view consisting of some simple attributes per grid cell like mean, minimum and maximum height (Hu and Yuan, 2016). Yang et al. (2017) also use 2D grids but improve the accuracy of ISPRS benchmark dataset by adding more geometric and full-waveform features to 2D grids. However, their method requires large memory to process the data.

Semantic point cloud segmentation can also be solved by 3D deep learning networks. To adapt unstructured point clouds to 3D convolutional filters, a group of methods partition 3D space into small grid voxels (Maturana and Scherer, 2015; Tchapmi et al., 2017; Wu et al., 2015). However, comparing to original point clouds, the voxelization not only causes loss in data representation but also introduces artifacts. These drawbacks hinder the learning of 3D features. Also, voxel structures store unoccupied grids in 3D space and this leads to high memory requirements. To avoid the disadvantages of the voxelization, networks that can directly consume unstructured point clouds are designed.

PointNet (Qi et al., 2017a) directly takes unstructured points as inputs and learns pointwise feature through a sequence of Multilayer Perceptron (MLP) layers. With the spatial transformer, the network is robust to variance in geometric transformation. Since PointNet only allows pointwise features to be learned independently, PointNet++ with a hierarchical structure is proposed to capture the geometric relationships among points in different scales (Qi et al., 2017b).

To exploit the contextual information among neighbouring points like 2D convolutional kernels, 3D convolutional networks are also introduced to directly consume irregularly structured point clouds. Unlike Voxnet (Maturana and Scherer, 2015) applying 3D convolutions on a discrete space, some methods define 3D convolutional operators over continuous space. For each point, weights of the neighbouring points depend on spatial distribution around the central points. Thomas et al. (2019) define both fixed and deformable Kernel Point Convolutions (KPConv) on continuous space. Linear correlation, assessing the distances between neighbouring points and kernel

points, is applied to assign different weights to different areas inside the domain of convolutional kernels. Positions of kernel points in deformable convolutions are learnable and can adapt to local geometry. Some other methods also define 3D operators in continuous space like SpiderCNN (Xu et al., 2018), PointConv (Wu et al., 2019) and Flex-Convolution (Groh et al., 2019).

Some researchers introduce graph convolutions to semantic point cloud segmentation tasks, where each point is taken as a graph vertex and edges are defined by relations among neighbouring points. For example, EdgeConv (Wang et al., 2019b) dynamically computes graphs not only on 3D spatial space but also on higher dimensional feature space, in order to capture the topological information in point clouds. In addition to construct graphs over single points, Landrieu and Simonovsky (2018) define graphs on superpoints which are geometrically homogeneous point sets, aiming to efficiently deal with large scale point clouds. Edges in SuperPoint Graphs represent the adjacency relationships between superpoints. Graph Convolutional Networks (GCNs) are applied to exploit the contextual information among shapes and object and this makes the GCN to consider a wider range of point clouds compared to point based GCNs.

3.2.2 Active learning

The objective of active learning is to sample data based on a calculated informativeness metric and maximize model performance with fewer labelled samples. How to evaluate the informativeness of samples is the main research question in active learning and this has been studied in the machine learning community for a long time. There are many ways to evaluate the sample informativeness for active learning. Uncertainty-based active learning criteria is probably the simplest and most commonly used (Settles, 2009). It selects samples that the 'model' is least certain about, like margin sampling and least confident sampling. This is a simple and direct method for probabilistic learning models (Settles, 2009). Density weighted methods query samples by assessing the intrinsic distribution and structure of the data and select samples are representative to the whole dataset, like Gaussian similarity (Zhu, 2005), divergence similarity (McCallumzy and Nigamy, 1998) and clustering (Xu et al., 2007). Expected change based methods estimate the influence of unlabelled samples on the current model. For example, Settles and Craven (2008) choose samples that make the largest change in the model by calculating the expected length of the gradient.

Recently, active learning has been incorporated with deep architectures in many studies. Gal et al. (2017) propose several

uncertainty-based metrics based on Bayesian CNNs for image classification. Monte-Carlo dropout technique is applied to approximate the Bayesian process in networks and produce probabilistic output. Then entropy sampling, variance sampling and maximizing mutual information are employed to assess the sample informativeness. Beluch et al. (2018) use ensembles of neural networks to evaluate the model dependent uncertainty in image classification tasks. Beluch et al. (2018) train all ensembles with the same network architecture and the same data but with different initialization weights. Data informativeness is evaluated by several metrics, including entropy, variation ratio and mutual information estimate. Besides evaluating the uncertainty, some studies combine expected change based methods with CNNs in image classification and object detection tasks like Otálora et al. (2017) and Brust et al. (2020).

In addition to image classification and object detection tasks, some efforts have been spent on solving point cloud related tasks by active learning strategies. Luo et al. (2018) propose a workflow to integrate higher order Markov Random Field (MRF) with active learning in order to efficiently assign pointwise labels to mobile LiDAR point clouds with limited labelling efforts. Assuming two nearby points are likely to share the same label, Luo et al. (2018) evaluate the neighbour-consistency during the sampling. That means, for a certain supervoxel, it is taken as a wrongly labelled sample if its predicted label is not the same as the label of its nearby manually annotated supervoxel. In this case, the 'incorrectly labelled' samples will be queried and manually annotated and then used to improve model performance in the next iteration. Although this work takes the advantage of interactions among neighbouring supervoxels and saves manual labelling by selecting optimal training supervoxel, taking MRF as the classifier still requires hand-crafted features which is less representative compared to deep learning features. Feng et al. (2019) propose a framework to integrate a state-of-the-art deep learning method with uncertainty-based active learning queries for 3D object detection in point clouds. They evaluate both aleatoric (data dependent) and epistemic (model dependent) uncertainty through Monte-Carlo dropout and deep ensembles techniques. With their active learning strategies, the model only needs 40% of the labelled data to achieve comparable accuracy when using all labelled data. Lin et al. (2020) integrate the deep learning network PointNet++ with two data dependent metrics for the semantic segmentation of ALS point clouds. However, the method does not take advantages of the knowledge learned from the previous stage, which makes the training process very time-consuming.

3.2.3 Incremental learning

An important step to make the active learning training more efficient is to retain the knowledge from previous tasks, i.e. training steps. Incremental learning is a method where new data are continuously added to existing training data in order to extend the knowledge of the current model. In the deep learning paradigm, classifiers and task-specific features are jointly learned. New samples cannot be simply added to update parameters as can be done in models like least-squares regression, because neural networks are non-convex and highly non-linear. To update model parameters in non-convex and highly non-linear networks, optimization techniques, such as gradient descent, are implemented to gradually refine model parameters to achieve global optima. In this scenario, simply updating models incrementally by only using new data is likely to make large changes in previously learned weights, forcing the model to adapt to new data. Therefore, its performance on the old data dramatically degrades (Kirkpatrick et al., 2017).

To maintain the performance on the old task, Castro et al. (2018) propose an end-to-end incremental learning strategy by combining distillation loss with cross entropy loss for image classification tasks. Distillation loss which is used to transfer information between different networks is adapted to maintain knowledge obtained from previous tasks and a cross entropy loss is used to learn from new data. Rusu et al. (2016) propose progressive neural networks where features acquired from old tasks are blocked to retain previous knowledge and new sub-networks are created to learn information from new data. In addition to enlarging the network, Kirkpatrick et al., (2017) propose the elastic weight consolidation (EWC) which penalizes on the difference between the new and old tasks. Brust et al. (2020) integrate incremental learning strategies with active learning criteria for object detection. The incremental learning is achieved by simple yet effective fine-tuning. After selecting new data by active learning criteria, newly labelled samples and old samples are assigned with different weights and are mixed. The model trained by the old data is updated to acquire information from new data by fine-tuning with those weighted samples. Similarly, Zhou et al. (2017) propose a framework to actively and incrementally fine tune CNNs for biomedical images. To our knowledge, there is no research that combines active and incremental learning with deep learning for semantic segmentation of point clouds.

3.3 Method

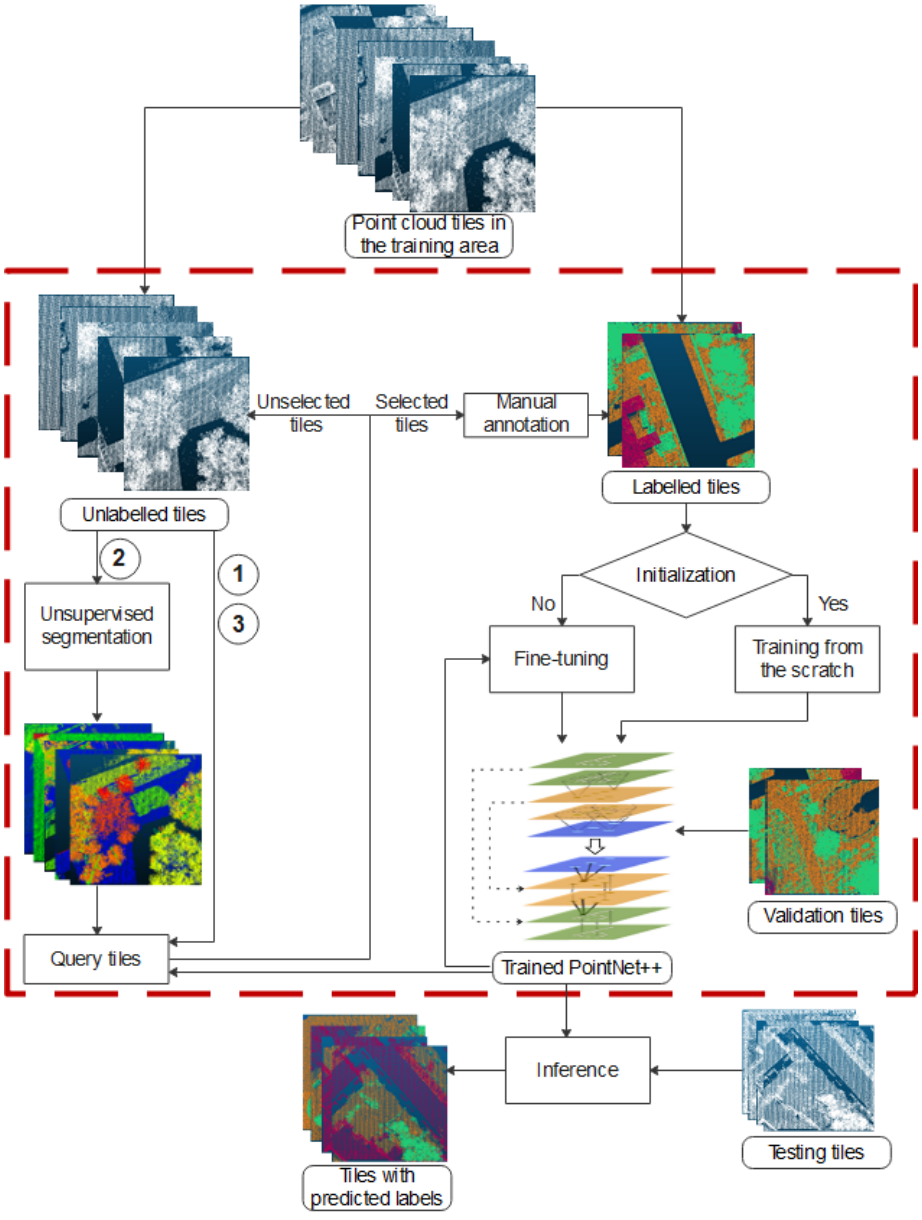


Figure 3.1 The proposed framework for active and incremental learning strategy for the semantic segmentation of point clouds. First of all, point clouds in the training area are split into tiles and separated into two groups: labelled (minority) and unlabelled (majority). If no

previous model is available, the network is trained from scratch. Otherwise, the model is incrementally fine-tuned according to the previous model. The model is validated on the validation tiles to avoid the overfitting to labelled data during the training. Then, the trained network selects unlabelled tiles by one of three queries depending on the metric to assess the informativeness of the tile. Query 1 directly consumes unlabelled tiles and query 2 relies on the unsupervised segmentation. Query 3 directly takes unlabelled points and evaluates the disagreement caused by model parameters. Selected tiles are labelled before the next training. The trained network is evaluated on testing tiles in each iteration.

The proposed workflow is presented in Figure 3.1. The red dash line box illustrates the active and incremental learning strategy introduced in this chapter. There are three major steps, namely, training, point cloud query, and annotation. The following sections first describe the active learning framework. Next, the details of the network used in this chapter are explained. Then, point entropy, segment entropy and mutual information are introduced to select informative point cloud tiles. Finally, how the incremental learning strategy is implemented in our research is explained.

3.3.1 Active learning

Let us consider a set of unlabelled point cloud tiles *S* which are generated by splitting the training area. To initialize the active learning framework, we first select and annotate a subset $\mathcal{L}_0$ from $\mathcal{S}$. We ensure $\mathcal{L}_0$ contains at least one instance of each class. Then $\mathcal{L}_0$ are excluded from $\mathcal{S}$ and we define the reduced unlabelled pool as $\mathcal{S}_0$. The initial network $\mathcal{M}_0$ is trained on $\mathcal{L}_0$. The active learning loop starts with the trained model $\mathcal{M}_0$ estimating the informativeness of all point cloud tiles in the unlabelled pool $\mathcal{S}_0$. Then we select K samples that are most informative and annotate them to form a set of labelled tiles $\mathcal{X}_1$. We update current labelled pool $\mathcal{L}_0$ with $\mathcal{X}_1$ to form a new labelled set $\mathcal{L}_1$ while excluding $\mathcal{X}_1$ from the current unlabelled pool $\mathcal{S}_0$ to form a new unlabelled set $\mathcal{S}_1$. Instead of training from scratch, we obtained a new model $\mathcal{M}_1$ by using all labelled tiles $\mathcal{L}_1$ to incrementally update the model $\mathcal{M}_0$ obtained from the previous step. For the n^{th} iteration, selected tiles, the labelled pool, the unlabelled pool, and the trained model are defined by $\mathcal{X}_n$, $\mathcal{L}_n$, $\mathcal{S}_n$ and $\mathcal{M}_n$ respectively. This querying and training loop is repeated until the stopping criterion is met, such as no significant improvement in network performance for several iterations or sufficient network performance has been achieved. Algorithm 3.1 summarizes the active and incremental learning strategy step by step.

Algorithm 3.1 Active Learning Algorithm

Input: a pool of unlabelled point cloud tiles $\mathcal{S}$

Output: the manually labelled point cloud tiles $\mathcal{L}$, and a trained neural network $\mathcal{M}$.

Initialization:

$\mathcal{L}_0 = SampleAnnotate\ (\mathcal{L})$ # Manually annotate a small subset.

$\mathcal{S}_0 = \mathcal{S} \backslash \mathcal{L}_0$ # Update set of unlabelled point cloud tiles.

$\mathcal{M}_0 = Train(\mathcal{L}_0)$ # Train $\mathcal{M}_0$ from scratch.

Active selection and incremental model updates:

$n = 1$

while the stopping condition is not met, do:

$\mathcal{X}_n = AL\ query(\mathcal{M}_{n-1}, \mathcal{S}_{n-1})$ # Select K tiles for labelling.

$\mathcal{L}_n = \mathcal{L}_{n-1} \cup \mathcal{X}_n$ # Update labelled tiles.

$\mathcal{S}_n = \mathcal{S}_{n-1} \backslash \mathcal{X}_n$ # Reduce unlabelled pool.

$\mathcal{M}_n = Train(\mathcal{M}_{n-1}, \mathcal{L}_n)$ # Use all labelled data to update the model.

$n = n + 1$

Return $\mathcal{L}_n$ and $\mathcal{M}_n$

Unlike picking points or super voxels proposed by Luo et al. (2018), point cloud tiles ($\mathcal{X}_n$) are queried by functions introduced in Section 3.3.2. Luo et al. (2018) calculate pre-defined pointwise geometrical features based on neighbouring points, like planarity and linearity. Then an MRF classifier is trained for assigning a label to each point according to those pre-defined features. In contrast, deep learning based methods learn geometrical features from data. That means the input of the networks should be a group of points that can preserve geometrical information, instead of a single point with a set of pre-defined features. Therefore, point cloud tiles are taken as the input of the forward pass and network weights are updated through back-propagation according to the loss function. If only a part of the tile is annotated, a complete tile is still required as the network needs the input to have geometrical representative charateristics. The computational cost will not change with the proportion of labelled points. The only thing that will be changed is the loss where unlabelled points give no contribution. If we query points from all unlabelled points, all point cloud tiles are required to be put into the network for each training. The training time for one epoch is the same as that of using the fully labelled whole training data. However, if we select point clouds by tiles, querying fewer tiles results in less computation time to complete a single training epoch.

3.3.2 Query functions

Four strategies to sample point cloud tiles are compared; random sampling, point entropy sampling, mutual information and segment entropy sampling. The random sampling picks tiles randomly and is taken as a baseline. We explain the other three methods in the following sections.

3.3.2.1 Point entropy

Shannon Entropy (SE) is an information metric indicating how much information is required to 'encode' a distribution.

$$E = -\sum_{c=1}^{C} prob(y = c|\boldsymbol{x}) \log prob(y = c|\boldsymbol{x}) \tag{3.1}$$

where $prob(y = c|\boldsymbol{x})$ is the predictive probability for class c coming after the softmax function at the last layer of the network.

When the model is quite certain about a class label, it will assign a very high predictive probability to that class and giving low values to other classes. In this case, the entropy value is low. On the contrary, high entropy value occurs when similar predictive probabilities are given to multiple classes and this suggests the model is not confident in the prediction. Here, we select samples that the model is most uncertain about and therefore those with high entropy values will be queried. In this research, point clouds are selected by tiles. To estimate the informativeness of the tile, we calculate the mean of pointwise entropy within each unlabelled tile. The K tiles with the highest average pointwise entropy will be queried, annotated, and added to labelled data pool for the next training.

3.3.2.2 Segment entropy

Apart from assessing pointwise uncertainty, the informativeness of point cloud tiles can also be evaluated at the segment level. The objective of point cloud segmentation is to partition point clouds into geometrically homogenous units. In this chapter, we use the unsupervised segmentation method proposed by Vosselman et al. (2017). This method combines both planar surface extraction algorithms and feature based segmentation methods. Firstly, a Hough transformation and surface growing algorithms are implemented to extract planar objects but they produce unnecessary small fragments in non-planar objects like trees. As a result, in the second step, only large segments are kept as planar objects and the remaining points are re-segmented by the feature based segment growing algorithm. The algorithm considers normal vector directions and planarity to group points on non-planar objects like vegetation, chimneys and cars.

Next, to overcome the over-segmentation on imperfect planar ground points, large adjacent segments are merged if their normal vectors are nearly parallel and points in one segment are also able to fit the plane of the other segment and vice versa. Finally, unsegmented points are given segment labels by majority voting in their neighbourhood. Isolated points still without a segment label are excluded from segment entropy calculation.

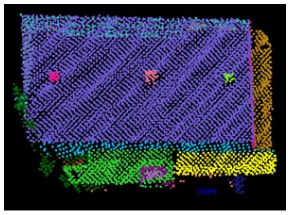
Here we assume that points within a geometrical homogenous unit share the same semantic label. Hence, if a model gives different labels to points within a segment, this model is likely to generate wrong predictions on this segment and thus those uncertain samples should be selected for the next training. The percentage of different predicted class labels within a segment is used to assess the informativeness of point cloud tiles. Suppose we have a segment consisting of N_{sp} points and predicted pointwise labels are represented by $[\hat{y}_1, \dots, \hat{y}_n, \dots, \hat{y}_{N_{sp}}]$. The following shows how we calculated segment entropy:

$$q_{ct}(c) = \frac{\sum_{n=1}^{N_{sp}} f_{ct}(\hat{y}_n, c)}{N_{sp}}, \quad \text{where } f_{ct}(\hat{y}_n, c) = \begin{cases} 1, & if\ \hat{y}_n = c \\ 0, & else \end{cases},\ \hat{y}_n = \text{argmax}_{y_n} prob(y_n|\boldsymbol{x}) \tag{3.2}$$

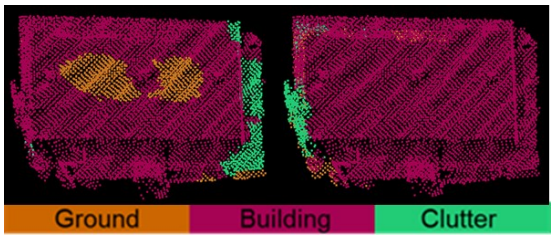
$$E_{seg} = -\sum_{c=1}^{C} q_{ct}(c) \log\, q_{ct}(c) \tag{3.3}$$

where E_{seg} represents the entropy within a segment and $q_{ct}(c)$ is the proportion of class c among the predicted labels, computed in equation (3.2). $\hat{y}_n$ is the predicted class label which has the highest predictive probability.

In order to avoid the underestimation of informativeness on large segments, segment entropy is given to their point members and then the mean of pointwise segment entropies is calculated to represent the informativeness of unlabelled tiles. Figure 3.2 illustrates the predicted labelled on the roof segment produced by models. The middle figure shows the variance of predicted labels on the roof and this variance leads to a high segment entropy. Tiles that comprise segments with high entropies are likely to be picked for the training in the next iteration.



A)



B)

Figure 3.2 Variation in predicted labels within a segment. A) Unsupervised segmentation results. Different colours represent different segments. B) Left: high entropy (0.465) within the roof segment. Right: low entropy (0.000) within the roof segment.

3.3.2.3 Mutual information

The above two metrics evaluate the data dependent (aleatoric) uncertainties. The following section explains how to estimate model dependent uncertainty by mutual information based on Bayesian Neural Networks. Bayesian Neural Networks are neural networks where prior probability distributions, like standard Gaussian priors, are placed over model parameters (Gal et al., 2017). However, direct inference from Bayesian networks is computationally expensive. Therefore, as a stochastic regularization technique, dropout which randomly ignores some of the neurons during the training, is used to approximate inference in Bayesian networks (Gal and Ghahramani, 2016). To extract uncertainty in prediction induced by the uncertainty in weights, multiple forward passes are performed with activated dropout during the testing, which samples from the approximate posterior.

In PointNet++, before the final prediction, a fully connected layer is inserted to integrate all features and then give logits to every class. Dropout is often set in this layer to prevent overfitting and is also used to construct a Bayesian network. Normally, we turn off the dropout during the prediction, but here we keep it on to get samples from the approximate posterior distribution of models.

Predictive probability distributions for N_{run} runs with dropout are represented by $p(y|x, w_1)$, $p(y|x, w_n)$, …, $p(y|x, w_{N_{run}})$. Mutual information

between predictions and model posterior is calculated by the following equation:

$$MI = \text{Entropy}(\frac{1}{N_{run}} * \sum_{n=1}^{N_{run}} p(y|x, w_n)) - \frac{1}{N_{run}} * \sum_{n=1}^{N_{run}} \text{Entropy}(p(y|x, w_n)) \quad (3.4)$$

where $\text{Entropy}()$ is the function of Shannon Entropy. High MI values suggest that the model is not confident in the predictions of samples on average, but different model parameters cause disagreement in predictions. In other words, each stochastic forward pass would have the highest predictive probabilities assigned to different classes. In this case, although the entropy of each run can be very small giving rise to a small value of the second term in equation (3.4), there is no significant large value in the averaged predictive probability distribution, which leads to the high entropy value for the model prediction, the first term in equation (3.4). Samples which maximize this MI metric are taken as informative data used for the next training. In our work, we calculate the average pointwise MI values within point cloud tiles and those causing uncertainties in model predictions are selected.

3.3.3 Semantic point cloud segmentation by neutral networks

An important component in our framework is the deep learning based model. Currently, many point based networks are available as mentioned in Section 3.2 and our proposed framework is supposed to be adapted to those models. To demonstrate the effectiveness of the proposed learning strategy, we pick PointNet++ (Qi et al., 2017b) as the model in this chapter. This is because Pointnet++ inherits MLP layers from PointNet to encode features in the local region, and the implementation of MLP layers which allow the network to directly consume points are still very popular in many deep learning based models (Landrieu and Simonovsky, 2018; Li et al., 2020; Wang et al., 2019b).

PointNet++ (Qi et al., 2017b) recursively apply a set of MLP layers to construct a hierarchical neural network. The network captures contextual information in point clouds by using set abstraction modules at multiple scales. The set abstraction module includes three sub-phases, namely, sampling, grouping and PointNet. A subset of the point cloud is collected by iterative farthest point sampling (FPS) in the sampling phase. The FPS gives better coverage of the point cloud and minimizing the clustering of points in a small region. This sampling strategy also adapts receptive fields to the points' distribution. In the grouping stage, neighbours around selected points are gathered. Then, selected points are taken as the input of the MLP. Here, a single input point relates to a small local region and represents a small point set,

in which each member contains its features, like XYZ coordinates or features obtained from previous set abstraction modules. In this chapter, the Adam optimiser (Kingma and Ba, 2014) is utilized to optimize the weights in PointNet++. A weighted cross entropy loss function is used to cope with imbalanced data. The dropout technique which ignores some of the neurons is implemented in the process of the training to avoid overfitting and we keep validating the model on the validation dataset to obtain optimal model weights. Early stopping is applied to terminate the training.

3.3.4 Incremental learning

Some of the active learning strategies focus on how to select samples step by step but ignore the knowledge learned from the previous learning stage during the training. For example. Luo et al., (2018) train the model from scratch for every step and Gal et al. (2017) train all models starting from a pre-trained VGG16 CNN model for the image classification task. To make good use of the previously learned information and speed up the training process, in this chapter, the model is incrementally fine-tuned from the model obtained in the previous step.

Most of the incremental learning methods mentioned in Section 3.2.3 deal with the case that new data are continuously added and this process may include new classes that are not used in the previous training. Also, old data is often unavailable in those cases and they can only train the model on the new data. However, in our study, the task is simpler. We do not introduce new classes during active learning steps and old data remain available. While keeping the model performance on all classes, we only need to make good use of the previous knowledge to speed up the training process instead of training from scratch for all models. Therefore, to speed up the training process, we modify the simple but effective strategy mentioned in Brust et al. (2020). Brust et al. (2020) suggest that parameters from the last active learning iteration can be used as the initialization of the current model in order to maintain the knowledge from previous training efforts. We incrementally fine-tune the models on both old tiles and newly selected tiles.

3.4 Experiments

Three active learning strategies are tested with ALS point clouds in our experiments. More details about the dataset, the specific structure of PointNet++, training parameters and how the proposed query functions are implemented are explained in the following paragraphs.

3.4.1 Data description

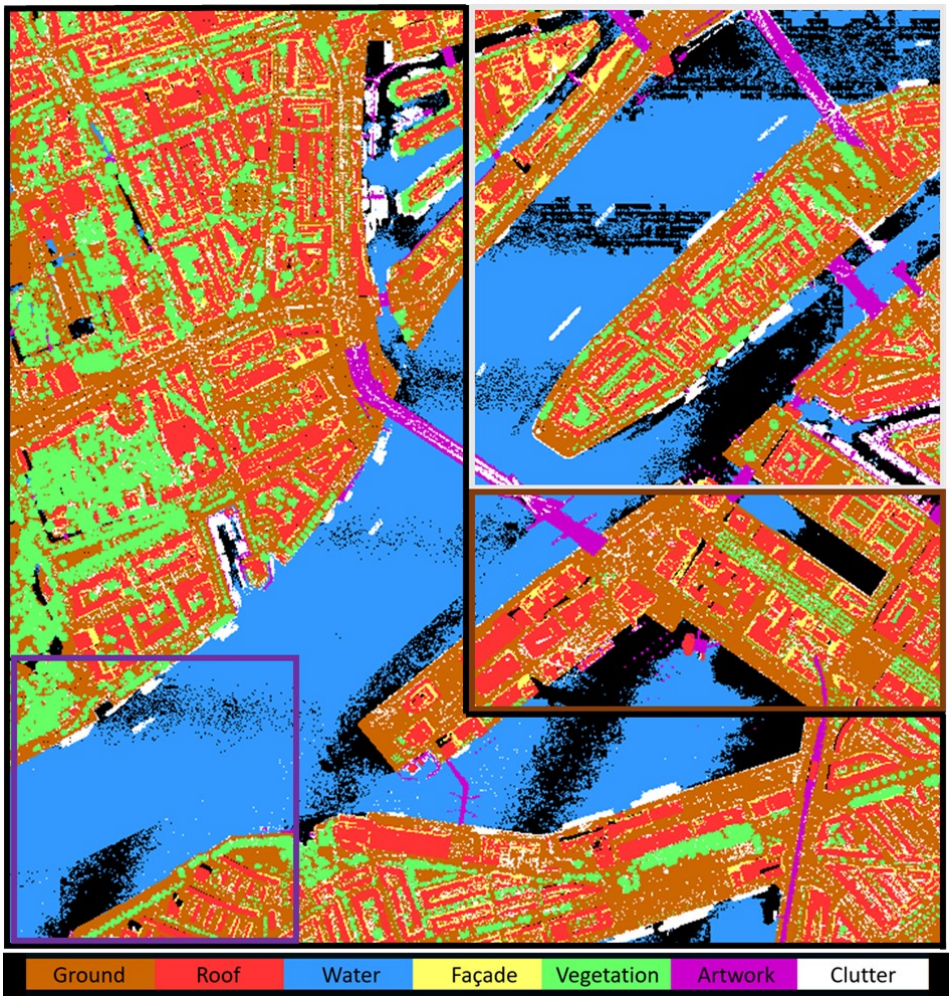


Figure 3.3 An overview of the study area in Rotterdam. The training area is in the black box. The validation area is in the brown box and the testing area is in the grey box. The area to initialize the model is in the purple box.

Actueel Hoogtebestand Nederland (AHN) dataset offers ALS point clouds with very high point density and high penetration from the multiple returns. It covers almost the entire area of the Netherlands. AHN3 is the latest version, covering more than half of the Netherlands. In this chapter, two subsets of AHN3 datasets are chosen for the experiments. The subsets are captured by an IGI LM6800 system with a 60° field of view. The mean strip overlap is 30% and the survey was

designed to obtain the point density at 60 points/m^2. One subset is located in the centre of Rotterdam (Figure 3.3), covering a 2 × 2 km^2 area. It is a densely built-up area with high rise buildings surrounded by trees and there are river channels with bridges. The point clouds were acquired on 4th December 2016 and manually annotated with seven classes, namely *ground*, *roof*, *water*, *façade*, *vegetation*, *artwork*, and *clutter*. The other dataset is situated in Amsterdam and was captured on 2nd February 2014. Its size is much larger than the Rotterdam datasets covering an area of 5 × 6.25 km^2 (Figure 3.4). The Amsterdam dataset not only includes the Amsterdam central area which is characterized by the buildings with complex shapes but also includes residential areas, parks and farmlands. Also, river channels in the Amsterdam dataset are crisscrossing and are much narrower than the river in the Rotterdam central area. As the Amsterdam dataset is quite large, we directly use the labels provided in the AHN3 dataset and classify points into 4 categories, namely *ground*, *building*, *water* and *clutter*.

3.4.2 Preprocessing

Since GPU memory is limited, it is unfeasible for a network to directly consume the whole study area. Therefore, the point cloud is cropped into 50 × 50 m^2 tiles and only XYZ coordinates are kept as the input of the network. We keep the Z-coordinates and normalize X- and Y-coordinates by the starting position of the tiles. In experiments, we randomly select 20,000 points as the input of the network. For tiles with more than 20,000 points, we select without replacement. For those with less than 20,000 points, all points are used as the input and the rest is compensated by random and repeated selection. With the purpose of making the model more robust to various orientations and noises, during the training, we randomly rotate point clouds around the Z-axis. Furthermore, Gaussian white noise with a σ of 4 cm is added to XYZ coordinates. These values are chosen empirically to add noise that will not significantly change the geometrical features for target objects.

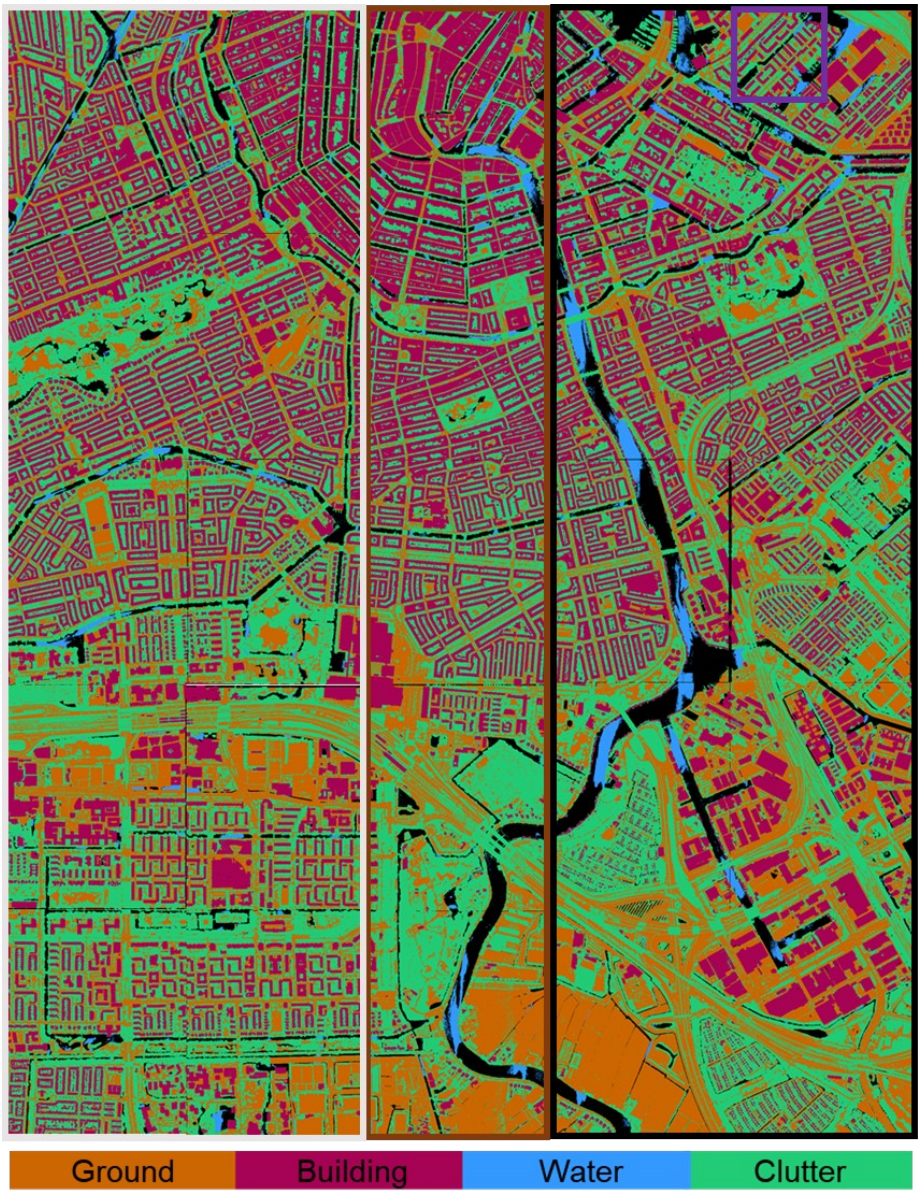


Figure 3.4 The overview of the Amsterdam dataset. The training area is in the black box. The validation area is in the brown box and the testing area is in the grey box. The area to initialize the model is in the purple box.

3.4.3 Network implementation

As mentioned in Section 3.3.3, PointNet++ consists of a sequence of sampling and grouping layers. Table 3.1 shows the spatial scales of set abstraction modules. The first sampling and grouping layer selects 4096 points from 20000 points in tile by iterative farthest point sampling strategy. Next, nearby points are grouped at two scales. 16 points are selected with a spherical search radius of 2 meters and 32 points are searched within 4 meters. For the next set abstraction module, 4096 points are subsampled to be 1024 and neighbouring points are searched and gathered within two larger scales. Fewer points can be sampled by the abstraction modules at higher levels and this inevitably leads to the loss in information in the latter layers of the network but this is beneficial for the network to exploit relationships among points in a wider range.

Table 3.1 Parameter configuration of multiple grouping modules in PointNet++

Level	Number of points	Search radius (m)	Number of neighbours
0	20000		
1	4096	[2, 4]	[16, 32]
2	1024	[4, 8]	[16, 32]
3	256	[8, 16]	[16, 32]
4	64	[16, 32]	[16, 32]

During the training, the learning rate for the initial model starts from 0.005 with a decay rate of 0.7 at every 75 training iterations. The learning rate keeps decreasing until it is less than 0.0001. Then the rate is kept at 0.0001 for the rest of the training. An early-stop strategy is applied to avoid overfitting. As the Rotterdam validation dataset is small, we check the performance on the validation dataset every epoch and stop training when the performance fails to improve over 15 epochs. For the Amsterdam dataset, the validation data contain about 2500 tiles. Here, it is not feasible to validate the model performance every epoch because the validation time is much longer than the training time. To balance the time spent on training and validation, we only check the model performance every several epochs. The check frequency changes with the size of the labelled data pool. Suppose we have J samples in the initial labelled training data, the check frequency is calculated by $\left\lceil \frac{2500}{J+K*n} \right\rceil$. Here K represents the number of newly added samples in the n^{th} iteration. The training is stopped when there is no improvement in model performance for 2 checks.

Due to the sampling, some points remain unlabelled in original point clouds. Therefore, pointwise predictions are propagated to the whole original tiles by nearest neighbour interpolation. To obtain the prediction on the test dataset, the data pass through the network 10 times and the predictive probability is averaged.

3.4.4 Accuracy assessment

Intersection over Union (IoU) (Everingham et al., 2010) is utilized to evaluate network performance. IoU per class is computed from true positives (TP), false negatives (FN) and false positives (FP) in confusion matrices as $\mathrm{TP} / (\mathrm{TP} + \mathrm{FN} + \mathrm{FP})$.

3.4.5 Active learning setup

For the Rotterdam dataset, an area where all seven classes (ground, roof, water, façade, vegetation, artwork, and clutter) exist should be selected to initialize the first model. The model performances initialized with different numbers of tiles ($\mathcal{L}_0$) are shown in Figure 3.5. Model performance is quite similar when $\mathcal{L}_0$ is set as 50 and 107. When $\mathcal{L}_0$ is 200, 410 tiles are required to achieve to the full training mIoU. In the following experiments, the area covering 600 × 600 m^2, consisting of 107 tiles is selected to initialize the first model. After excluding very sparse tiles, 783 tiles are in the unlabelled pool, waiting to be selected.

Then the next question is how many tiles we need to select in each iteration, that is the value of K mentioned in 3.3.2. Some studies only select a single sample in each iteration. As this requires to run the training and selecting process many times, it would be a very time-consuming process. Therefore, multiple point cloud tiles are queried in every iteration. Yet, it is not a wise choice to query a large portion of data like a quarter of the tiles for annotation because less important tiles which can only make little contributions to model performance will be selected and all tiles will be annotated in four iterations which conflicts with the purpose of the active learning, namely saving annotation efforts.

Figure 3.6 below illustrates how the model performance changes with increasing training tiles when selecting different numbers of tiles in each iteration. We test three sizes 10, 35 and 70 which corresponding to about 1%, 4% and 8% of the tiles in the training area respectively. When adding 35 tiles each iteration, after training and querying for 2.9 hours, 282 tiles are selected and fed to the model and the model performance becomes stable and fluctuated around the mIoU obtained by the full trained model. When taking K as 70, although the training and selecting process takes 2.5 hours, the model requires 387 annotated tiles to achieve the same status. When K equals 10, the

model takes 19 steps (6.9 hours) with 297 annotated tiles to reach the full train IoU. It fails to reduce the number of required tiles and leads to longer training time. Therefore, the labelled training pool is updated by 35 tiles in each iteration.

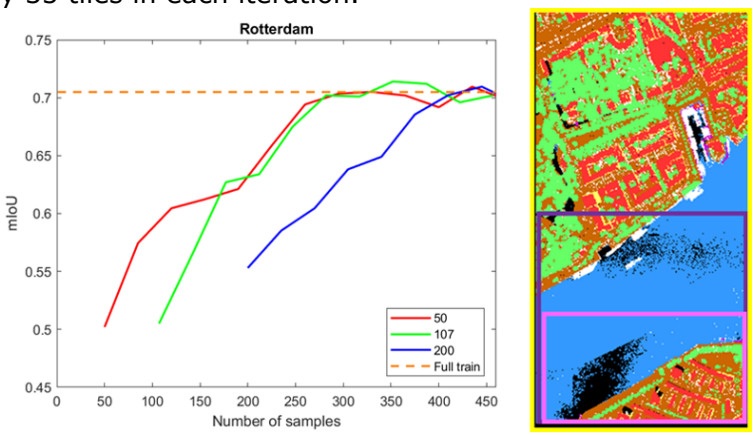


Figure 3.5 Comparison of model performance on the Rotterdam dataset with different sizes of the initial tiles (left). Point entropy is taken as the query function. The pink, purple and yellow boxes represent 300 × 600 m2, 600 × 600 m2 and 1200 × 600 m2 areas (right). After excluding very sparse tiles with 50, 107 and 150 tiles are left for training.

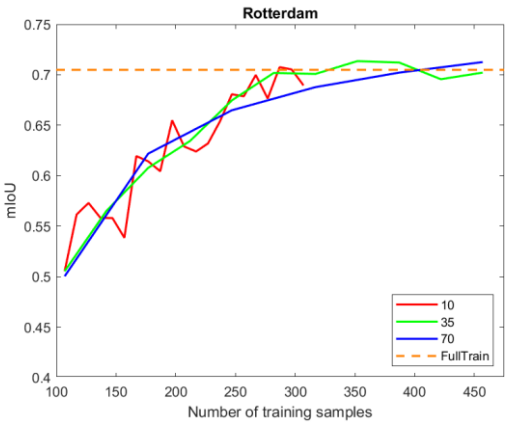


Figure 3.6 Comparison of the model performance with three sizes of selected point cloud tiles in each iteration in the Rotterdam dataset, using point entropy function.

For the Amsterdam dataset, the model performances initialized with different numbers of tiles ($\mathcal{L}_0$) are shown in Figure 3.7. When the model is initialized by 200 tiles, it requires more than 600 labelled point cloud tiles to reach the full training mIoU. Here we select a 500 × 500 m^2

area, which contains 100 tiles to initialize the first model. We compare three sizes of queried tiles, namely 50, 100, 200 which corresponding to about 1%, 2% and 4% of the tiles in the training area respectively (Figure 3.8). It can be seen that K equals 50 firstly approach the around the mIoU obtained by the full trained model while the other two values still require more iterations. To save manual annotation efforts, the training data is updated by 50 tiles step by step.

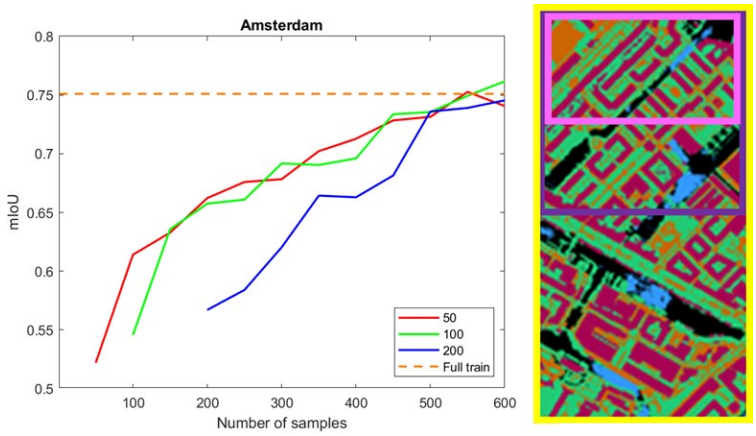


Figure 3.7 Comparison of model performance on the Amsterdam dataset with different sizes of the initial tiles. Point entropy is taken as the query function (left). The pink, purple and yellow boxes represent 250 × 500 m2, 500 × 500 m2 and 1000 × 500 m2 areas (right), corresponding to 50, 100 and 200 tiles.

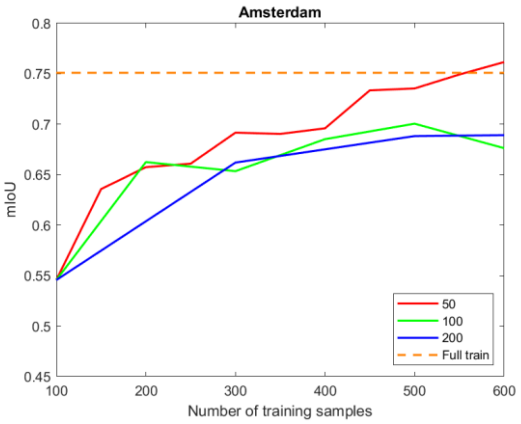


Figure 3.8 Comparison of the model performance with three sizes of selected point cloud tiles in each iteration in the Amsterdam dataset, using point entropy function.

In our experiments, the active learning strategies based on point entropy, mutual information and segment entropy are compared with the baseline method in which unlabelled tiles are randomly selected. For the mutual information metric which evaluates disagreements among various model variants, each point cloud tile is predicted under 10 different parameter settings. The process of querying and training is run for 10 iterations to see which strategy first makes the model achieve a high level with the least training samples. To demonstrate the effectiveness of the proposed method, for each query, experiments are repeated 3 times and results are averaged.

3.4.6 Incremental learning setup

In our experiments, we incrementally fine tune models with all available data. Although fine-tuned with only newly selected data takes less time to train the model (Table 3.2), its model performance is much worse compared with using all available data.

To set the learning rate for fine-tuning, the effects of the learning rate on model performance are presented in Figure 3.10 and computation times are listed in Table 3.3. The active and incremental learning process ends up with similar model performance with different fine-tuning learning rates. However, when the learning rate is 0.005, the model takes a longer time to converge when it is compared to the other two values. Here we set the fine-tuning learning rate as 0.0001 to avoid models fall into local optima.

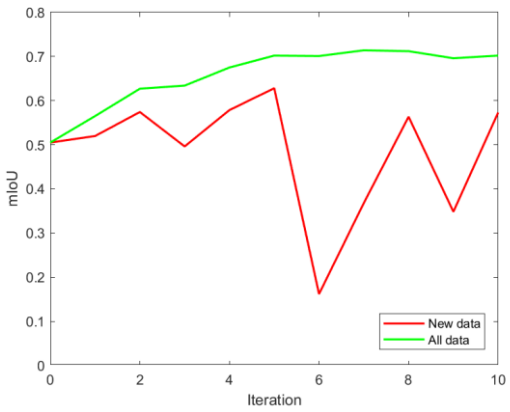


Figure 3.9 Comparison of model performance on the Rotterdam dataset when models are incrementally fine-tuned with only newly selected data (New data) and all available data (All data). Here we use point entropy as the query function.

Table 3.2 Comparison of required training time for the Rotterdam dataset when models are incrementally fine-tuned with only newly selected data (New data) and all available data (All data). Here we use point entropy as the query function.

Method	New data	All data
Training time (hours)	2.6	3.8

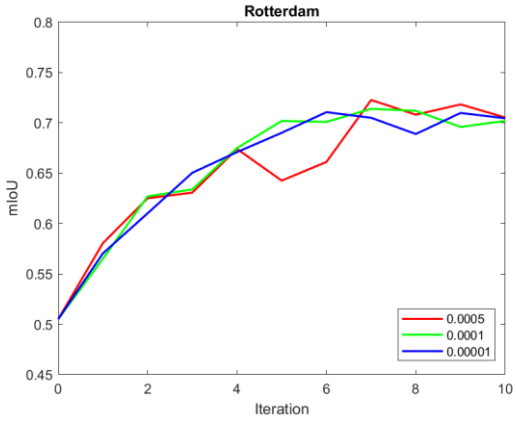


Figure 3.10 Comparison of the model performance on the Rotterdam dataset for different learning rates used in fine-tuning. Point entropy is taken as the query function.

Table 3.3 Comparison of the computation time required for the Rotterdam dataset using different fine-tuning learning rates.

Learning rate	0.0005	0.0001	0.00001
Training time (hour)	4.40	3.80	3.71

3.4.7 Results

3.4.7.1 Comparison of selection queries

Model performances of various active learning functions are presented in Figure 3.11 and Figure 3.16.

3.4.7.1.1 Rotterdam

Figure 3.11 and Table 3.4 illustrate how the model performance on the Rotterdam dataset changes with an increasing number of samples selected by different active learning query functions. It can be seen that for all functions model performance tends to increase with some fluctuations. Point entropy, segment entropy and mutual information give better results than the baseline method. The random selection leads to an unstable model performance which is illustrated by the

large standard deviation in the mIoU. When it comes to the other three query functions, the standard deviation is only large at the beginning which can be explained by the asynchronous improvements among different runs. Then the standard deviation becomes relatively small in the later iterations where mIoU is similar to the value obtained from the full training. This suggests that the selected samples do provide useful information to improve model performance. When comparing three query functions, segment entropy performs the best and it first reaches full training accuracy at the fifth iteration where 282 tiles, 31.7% of the tiles in the training area are used. Point entropy also reaches full train mIoU at the fifth iteration but its mIoU is a little bit lower than that of segment entropy. Mutual information reaches the full train mIoU at the 6th iteration where 35.6% of the tiles in the training area are used. In terms of mIoU, all query functions select meaningful data for model training and can be used to save manual annotation efforts.

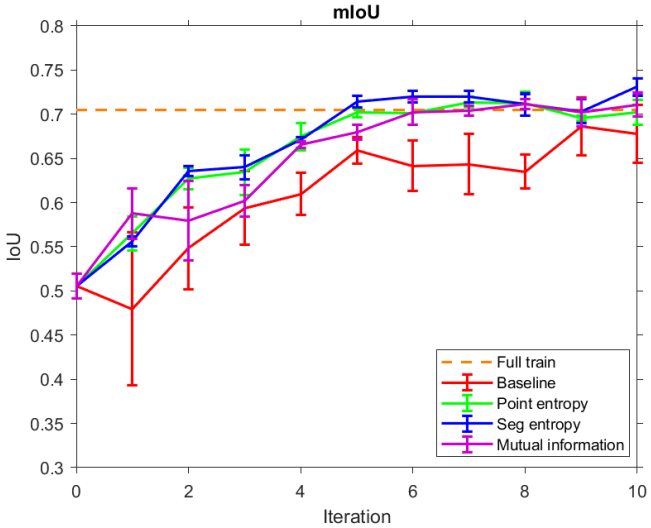


Figure 3.11 Mean IoU scores of baseline and active learning strategies with different query functions for the Rotterdam dataset. The horizontal axis represents the iteration. Error bars represent standard deviations.

Figure 3.12 shows the change in IoU for different classes (lines) and the variation of training data distribution (columns) with more selected training samples. It can be seen that the three query functions improve model performance for the classes ground, water, clutter, and artwork. For other classes, the IoU values are comparable to those of the baseline but the results are more robust as the standard deviations are much smaller than those of the baseline. One possible reason for the insignificant improvement in these classes is that they are relatively

easy classes for the model and the model can easily acquire enough information to differentiate them and then reach high accuracy. As a result, newly added samples are hard to enrich the model knowledge of these classes. One observation is that all three query functions are able to select tiles with difficult classes like artwork and clutter. When selecting samples by our query functions, the IoU values for artwork and clutter are higher than those of the baseline, especially artwork. This suggests that all active query functions achieve the objective of selecting informative samples for deep learning models.

Table 3.4 Mean IoU scores of baseline and active learning strategies with different query functions for the Rotterdam dataset. For every function, the first value to reach the full train mean IoU is in bold.

Iteration	0	1	2	3	4	5	6	7	8	9	10
Baseline	0.505	0.480	0.548	0.594	0.610	0.659	0.642	0.643	0.635	0.686	0.678
Point entropy	0.505	0.565	0.627	0.634	0.675	0.702	0.701	**0.714**	0.712	0.696	0.702
Mutual information	0.505	0.588	0.580	0.602	0.666	0.680	0.703	0.704	**0.712**	0.702	0.711
Segment entropy	0.505	0.556	0.636	0.640	0.671	**0.714**	0.720	0.720	0.711	0.703	0.731

Figure 3.13 demonstrates some samples selected by point entropy uncertainty. It can be seen that high uncertainty values are around object boundaries, clutter objects and on sloped ground. The model is uncertain on points within slanted ground segments because those segments are similar to slanted roofs in terms of the geometry. However, this uncertainty is not visible for segment entropy because the segmentation algorithm separates flat ground and slant ground into different parts. Most of the sloped ground points are predicted to be roof leading to a low value in segment entropy.

Figure 3.12 shows that mutual information selects tiles with abundant tree points which are also shown in Figure 3.14. Although most of the vegetation and ground points are correctly predicted, the mutual information is still high on the ground points because with dropout during the testing, some models are quite confident in predicting ground points as vegetation. However, selecting tiles that only have ground and vegetation points makes little contribution to model knowledge because vegetation and ground points already got relatively high accuracy. Similar to segment entropy, mutual information can also detect the uncertainty at object boundaries.

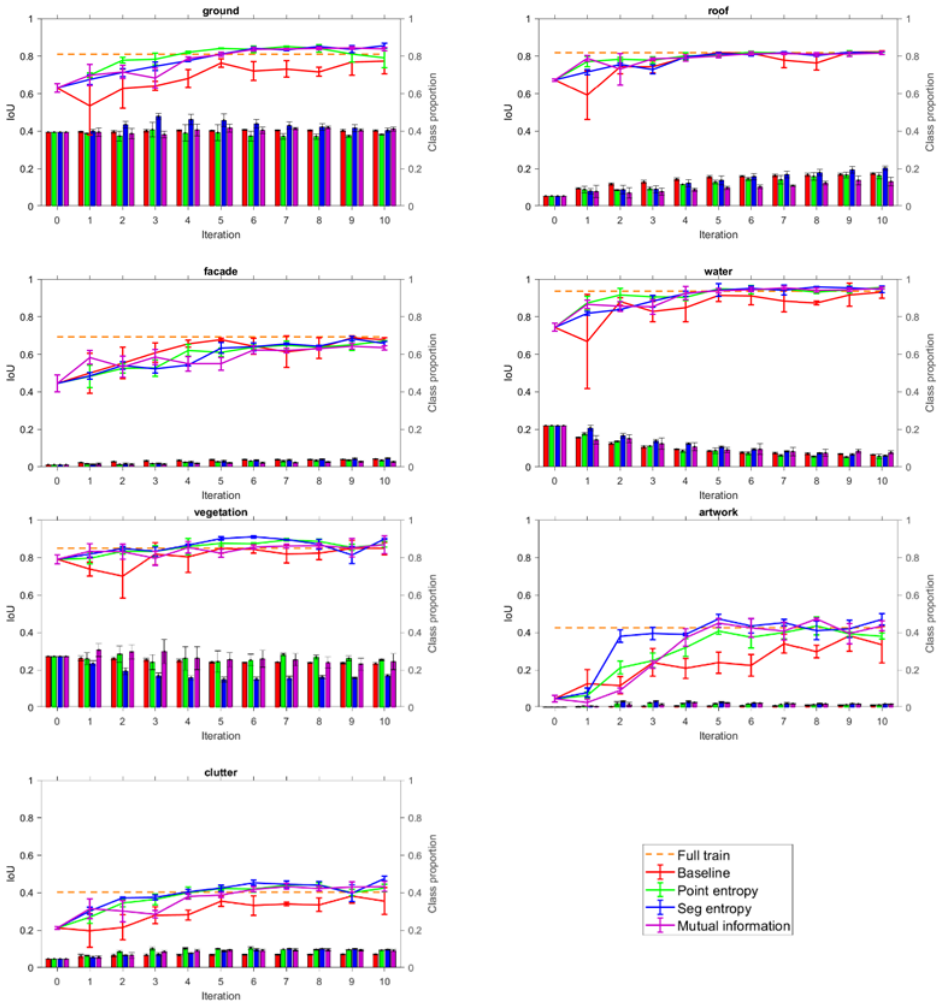


Figure 3.12 IoU (lines) and data distribution (columns) for different classes in the Rotterdam dataset. The horizontal axis represents the iteration. Error bars represent standard deviations.

Relating to the distribution columns in Figure 3.12, segment entropy prefers tiles with more ground points comparing to point entropy and mutual information. For example, in Figure 3.15, PointNet++ is not good at object boundaries and some ground points surrounding clutter objects are predicted as clutter. As we enforce the consistency within segments, those wrongly predicted flat ground points enlarge the segment entropy of tiles. Although its IoU values in ground are not better than the other two methods, this could help solve the confusion between clutter and ground points and explain the better accuracy for clutter (Figure 3.12). Also, segment entropy selects scenes where trees are quite close to building facades and part of the canopy is likely to

be predicted as façade points. As a tree canopy is always taken as one segment according to the unsupervised segmentation algorithm, the inconsistency of predicted labels in a canopy segment leads to large segment entropy over the tile.

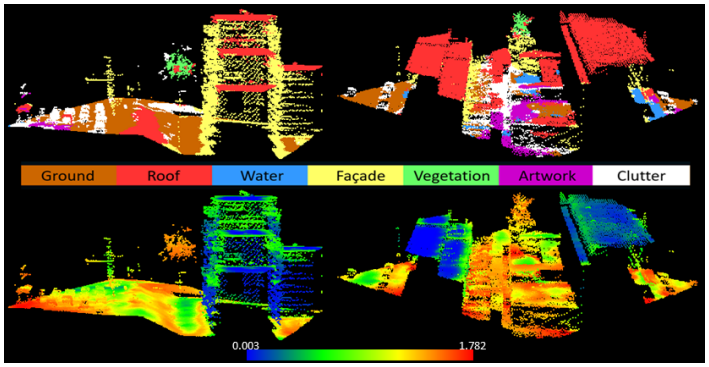


Figure 3.13 Example of tiles selected by point entropy. The first row shows the predicted label. The second row shows the corresponding pointwise point entropy values.

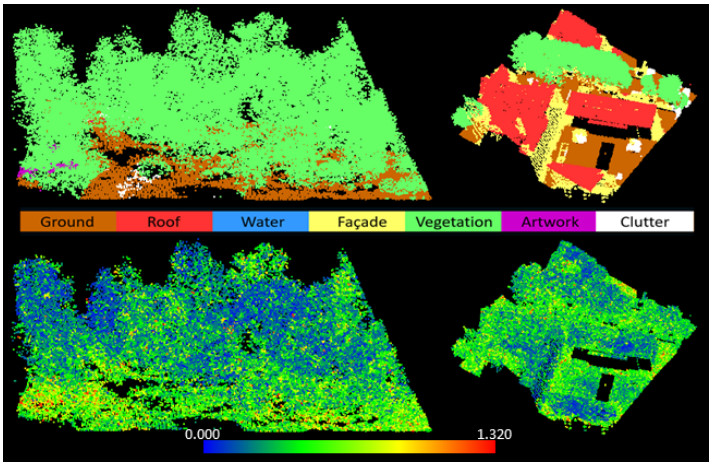


Figure 3.14 Example of tiles selected by Mutual information. The second row shows the corresponding pointwise mutual information values.

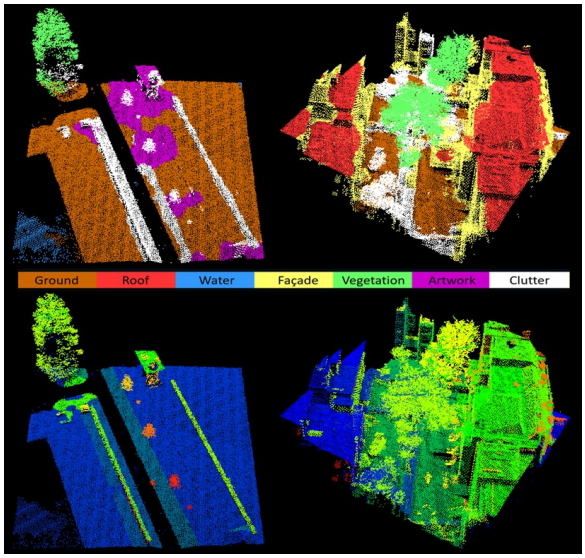


Figure 3.15 Example of tiles selected by segment entropy. The first row shows the predicted label. The second row shows the corresponding unsupervised segmentation results.

Table 3.5 demonstrates the computational time to train models and query samples for the Rotterdam dataset. The table indicates that the training times required for different methods are quite similar but the mutual information metric takes much longer time to select informative point cloud tiles. This is because as mentioned in Section 3.3.2.3, point clouds are supposed to pass the network multiple times, in order to estimate the disagreement among the predictions. In comparison, point entropy and segment entropy are much faster to select informative samples.

Table 3.5 Comparison of the computational time required for the Rotterdam dataset using different query functions.

	Baseline	Point entropy	Mutual information	Segment entropy	Full train
Training (hour)	4.04	3.80	4.18	3.94	0.97
Querying (hour)	0.00	1.17	8.50	1.20	0.00
Sum (hour)	4.04	4.97	12.68	5.14	0.97

3.4.7.1.2 Amsterdam

Figure 3.16 and Table 3.6 show how the model performance on the Amsterdam dataset responds to an increasing number of samples selected by different active learning query functions. In terms of the

mIoU, the performance of the baseline is quite stable after the third iteration. The improvement in model performance is insignificant with an increasing number of randomly selected tiles and the mIoU is much lower than that of the other three methods at the final iteration. For the other three query functions, the mIoUs gradually increase to a level which is slightly above the mIoU achieved by using all samples during the training. Segment entropy firstly reaches the full-train mIoU at the seventh iteration where only 9% of the tiles are used for training. Mutual information achieves 0.759 (mIoU) using 10% of the tiles at the eighth iteration and point entropy only reaches 0.762 (mIoU) with 12% of the tiles at the tenth iteration. Computational times to train models and query samples for the Amsterdam dataset are presented in Table 3.7. Except mutual information, point entropy and segment entropy take over 9 more hours compared to full training while it significantly reduces annotation efforts which is more time-consuming.

When analysing the IoU for each class in Figure 3.17, it can be seen that the IoU values for the classes ground, building and clutter are quite similar for all strategies and the baseline method even has a higher IoU in clutter and building before the sixth iteration. The main differences lie in the IoU for the water where the lines for the three query functions are above the line for baseline. Unlike the Rotterdam dataset where the river is wide and easy to be recognized, the identification of water points by PointNet++ is challenging in the Amsterdam dataset, because the canals in the dataset are crisscrossing and narrow. Our query functions select tiles with more water points and contribute to higher IoU values for the water. This is similar to the selection of more tiles with artwork and clutter improved the performance in the Rotterdam dataset. This suggests that selected samples enrich the model knowledge in difficult classes.

Figure 3.18 presents the spatial distribution of all selected tiles according to different active learning strategies. Selected tiles for the baseline method are randomly spread over the training area while the other three queries select tiles located in the southern part of the training area which is dominated by farmlands, different from the densely built-up area in Amsterdam centre. The knowledge of farmlands brings slight advantages over the baseline in the IoU for ground before the third iteration.

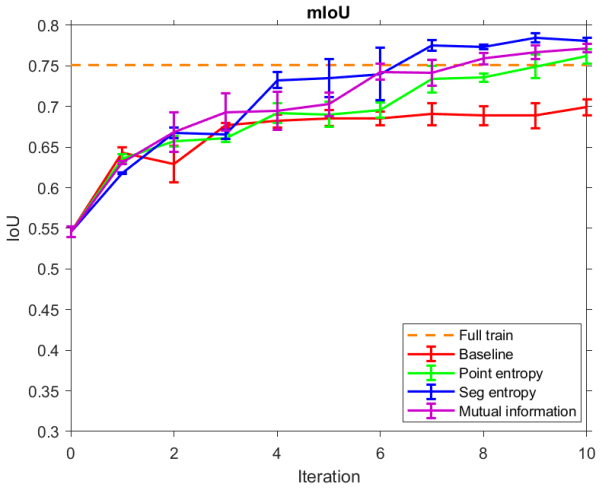


Figure 3.16 Mean IoU scores of baseline and active learning strategies with different query functions for the Amsterdam dataset. The horizontal axis represents the iteration. Error bars represent standard deviations.

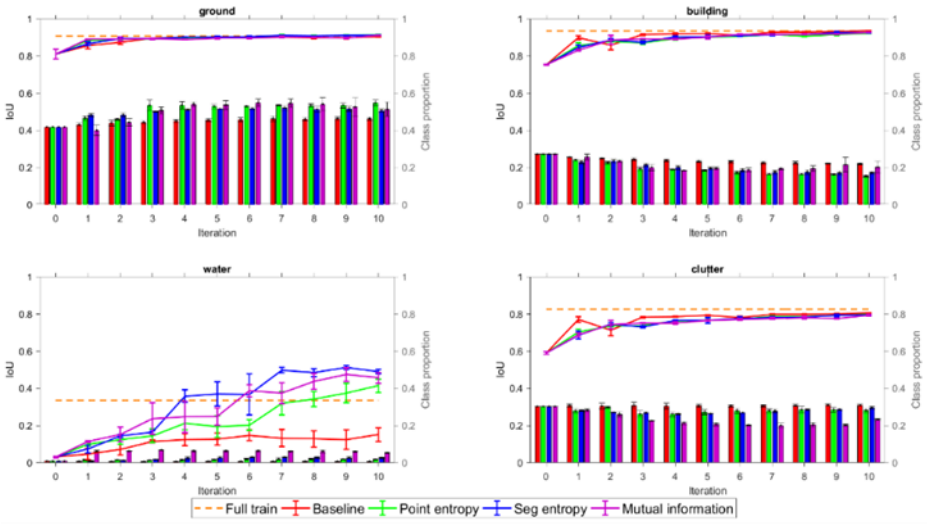


Figure 3.17 IoU (lines) and data distribution (columns) for different classes in the Amsterdam dataset. The horizontal axis represents the iteration. Error bars represent standard deviations.

Table 3.6 Mean IoU scores of baseline and active learning strategies with different query functions for the Amsterdam dataset. For every function, the first value to reach the full train mean IoU is in bold.

Iteration	0	1	2	3	4	5	6	7	8	9	10
Baseline	0.546	0.643	0.629	0.677	0.682	0.685	0.686	0.691	0.689	0.689	0.699
Point entropy	0.546	0.636	0.657	0.661	0.692	0.690	0.696	0.734	0.735	0.749	0.762
Mutual information	0.546	0.631	0.668	0.693	0.695	0.703	0.743	0.742	0.759	0.767	0.772
Segment entropy	0.546	0.618	0.668	0.666	0.732	0.735	0.740	0.775	0.773	0.785	0.780

Table 3.7 Comparison of the computation time required for the Amsterdam dataset using different query functions.

	Baseline	Point Entropy	Mutual information	Segment entropy	Full train
Training time (hour)	10.11	10.33	9.35	10.16	4.78
Querying (hour)	0.00	3.84	30.65	4.15	0.00
Sum (hour)	10.11	14.17	40.00	14.31	4.78

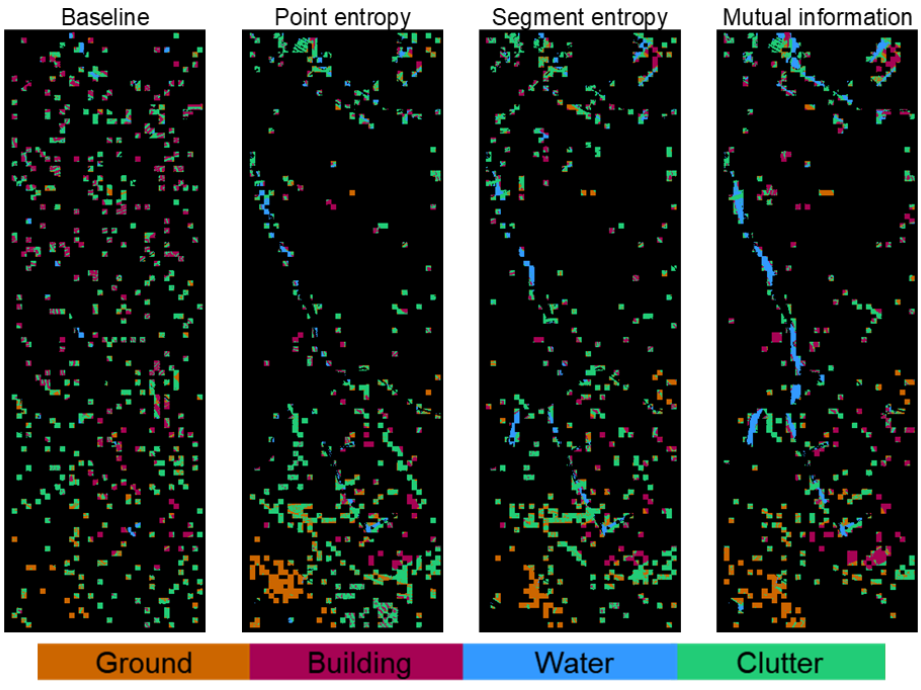


Figure 3.18 Spatial distribution of selected tiles in Amsterdam.

3.4.7.2 Comparison between fine-tuning and training from scratch

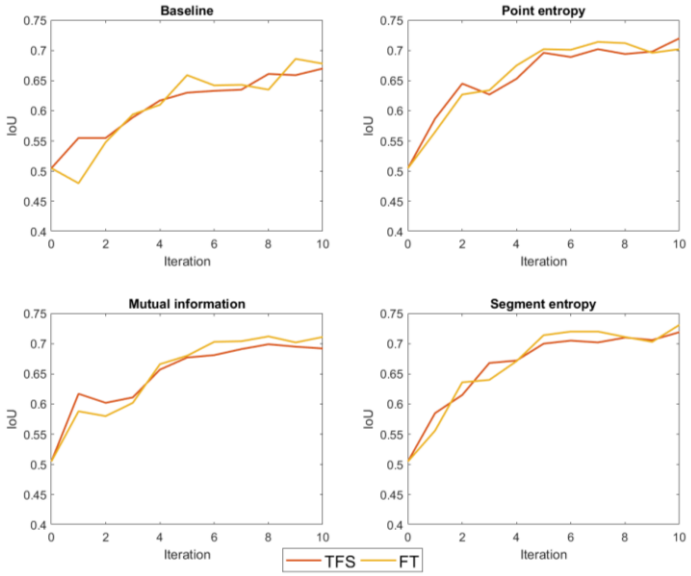


Figure 3.19 Comparison of model performance on the Rotterdam dataset under different training strategies, training from scratch (TFS) and fine-tune (FT). The vertical axis represents the mIoU and the horizontal axis represents the iteration.

Figure 3.19 shows the model performance on the Rotterdam dataset when training from scratch and fine-tuning the model from the previous model. It can be seen that fine-tuning can achieve the accuracy comparable to training from scratch but it effectively saves training efforts.

Figure 3.20 shows how many updates the model requires during the training. More updates mean a longer time for training. When using all tiles in the training area, the batch accuracy gradually increases with fluctuations which are caused by the randomness of point cloud tiles in each batch. When all models are trained from scratch in each active learning iteration, the batch accuracy drops dramatically because no previous knowledge is involved and it takes some updates for the model to learn. While fine-tuning requires more updates compared to full training for all active learning strategies, it saves about half of the training efforts comparing to the training from scratch. It can be seen that the model keeps the previous knowledge and avoids low batch

accuracy at the beginning of the training in each iteration. Table 3.8 demonstrates the training time required for training from scratch, fine-tuning and full training. Incrementally fine-tuning is much faster than straining from scratch.

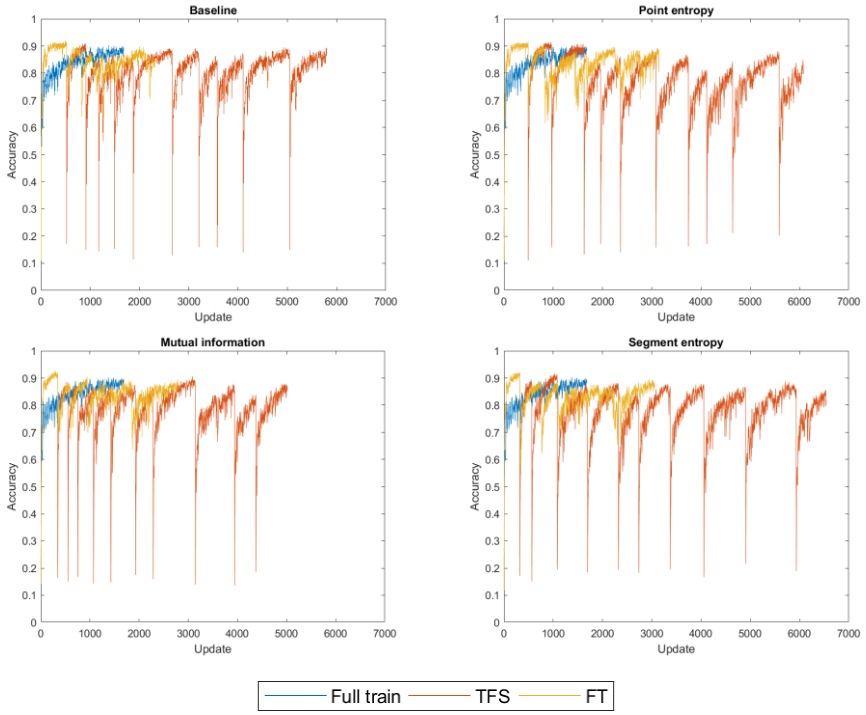


Figure 3.20 Accuracy (batch accuracy) for each update during the training. During the training, a batch of samples is randomly drawn from the entire labelled data to update the model parameters by stochastic gradient descent. Here 'update' means updating the network weights using 16 point cloud tiles. For each tile, 20,000 points are randomly selected. Here the (batch) accuracy represents the number of correctly (predicted points) / (20,000*16). For training from scratch and fine-tuning, we accumulate the number of performed updates and the training accuracy of each update is plotted from the initial training to the 10th training. (Full train: use all tiles in the training area in once. TFS: for each step, the model is trained from scratch. FT: for each step, the model is fine-tuned from the previous model.)

Table 3.8 Comparison of the training time required for the Rotterdam dataset using different training strategies. (TFS: for each step, the model is trained from scratch. FT: for each step, the model is fine-tuned from the previous model.)

	Baseline	Point entropy	Mutual information	Segment entropy
TFS (hour)	6.96	6.62	6.35	6.49
FT (hour)	4.04	3.80	4.18	3.94
Full train (hour)	0.97			

3.5 Conclusion

Existing supervised deep learning networks for semantic point cloud segmentation require a large number of labelled points for training. This research proposes an active and incremental learning workflow to effectively reduce annotation efforts by iteratively selecting informative samples and incrementally enriching the model knowledge. Firstly, point clouds are split into tiles and separated into labelled and unlabelled groups. Then the labelled tiles are used for training. For the initial iteration, the network is trained from scratch. For the rest of the steps, fine-tuning is implemented to incrementally enlarge the model knowledge based on the previous model. In each iteration, after the training, the informativeness of point cloud tiles in the pool of unlabelled training tiles is evaluated by the trained network according to three uncertainty metrics, namely point entropy, segment entropy and mutual information. Both point entropy and segment entropy assess the data dependent uncertainty while the segment entropy considers the interactions among neighbouring points within geometrical homogenous units. Mutual information, which estimates the model dependent uncertainty, is derived from Bayesian networks. The idea is to analyse the disagreements in model predictions caused by the uncertainty of model parameters. The most informative tiles are labelled and added to the labelled training pool for the next training.

The framework is tested on two subsets of AHN3 datasets. Experimental results show that compared to the random selection, all three metrics are capable of selecting informative point clouds like tiles dominated by difficult classes and samples diversifying geometry of target objects in the labelled training pool. Among the three query functions, segment entropy performs the best. For the 7 class classification in the Rotterdam dataset, it takes 31.7% of the whole training area to reach the mIoU obtained from the model trained on the whole training area. When it comes to the 4 class classification in the Amsterdam dataset, it only requires 9% of the whole training area

to achieve the full training mIoU. Also, the effectiveness of incremental learning is verified on the Rotterdam dataset. It saves about half of the training efforts comparing to the training from scratch for each active learning iteration.

The proposed framework is successfully tested with two ALS datasets using PointNet++. Although we perform experiments on PointNet++, the three uncertainty metrics can also be applied to many other state of the art network architectures. Point entropy requires the predictive probability for each class and mutual information needs to turn on dropout during the testing. These conditions can be easily met by most of the networks like PointCNN, KPconv and SPG. However, segment entropy, evaluating the interactions among points within segments, can only be applied to point based networks. It is invalid for segment based method, like SPG, where segments are taken as homogenous units and points within a segments share the same label. In addition to ALS data, the framework is also possibly generalized to point clouds from terrestrial mobile laser scanners or indoor scenes.

Chapter 4 – Weakly Supervised Semantic Segmentation of Airborne Laser Scanning Point Clouds [3]

[3] This chapter is based on:

Lin, Y., Vosselman, G., Yang, M.Y., 2022. Weakly supervised semantic segmentation of airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 187, 79–100.

Abstract

While modern deep learning algorithms for semantic segmentation of airborne laser scanning (ALS) point clouds have achieved considerable success, the training process often requires a large number of labelled 3D points. Pointwise annotation of 3D point clouds, especially for large scale ALS datasets, is extremely time-consuming work. Weak supervision that only needs a few annotation efforts but can make networks achieve comparable performance is an alternative solution. Assigning a weak label to a subcloud, a group of points, is an efficient annotation strategy. With the supervision of subcloud labels, we first train a classification network that produces pseudo labels for the training data. Then the pseudo labels are taken as the input of a segmentation network which gives the final predictions on the testing data. As the quality of pseudo labels determines the performance of the segmentation network on testing data, we propose an overlap region loss and an elevation attention unit for the classification network to obtain more accurate pseudo labels. The overlap region loss that considers the nearby subcloud semantic information is introduced to enhance the awareness of the semantic heterogeneity within a subcloud. The elevation attention helps the classification network to encode more representative features for ALS point clouds. For the segmentation network, in order to effectively learn representative features from inaccurate pseudo labels, we adopt a supervised contrastive loss that uncovers the underlying correlations of class-specific features. Extensive experiments on three ALS datasets demonstrate the superior performance of our model to the baseline method.

4.1 Introduction

Airborne laser scanning (ALS) point clouds are important data sources for many products like digital terrain models (DTM) (Chen et al., 2017), 3D city models (Zhou et al., 2020) and landscape models (Murtha et al., 2018). These products are required in a wide range of applications like land administration (Lemmen et al., 2015), urban planning (Murgante et al., 2009) and disaster management (Shen et al., 2010). Point cloud interpretation is an essential step to produce these products and semantic segmentation is one kind of interpretation where each point in point clouds is assigned a semantic label. However, manually assigning a label to every single point requires a huge amount of manual efforts, especially for large scale datasets. Therefore, many machine learning algorithms are proposed to automate the pointwise interpretation.

Semantic segmentation of point clouds through machine learning techniques has been investigated, from extracting representative hand-crafted features (Lin et al., 2014; Weinmann et al., 2013) to using different classifiers (Chehata et al., 2009; Lodha et al., 2007, 2006; Xu et al., 2014). In addition, in order to capture contextual information among adjacent points, some methods use graphical models like Markov Random Fields (MRF) (Najafi et al., 2014), and Conditional Random Fields (CRF) (Niemeyer et al., 2014) or a versatile regularization framework (Landrieu et al., 2017). Recently, deep learning approaches have been proven to be an effective solution that do not rely on predefined hand-crafted features but can learn highly representative features from the data. Many algorithms have been developed for the semantic segmentation of point clouds and they can be roughly categorized into three branches according to their input data structures, namely projection based (Boulch et al., 2018; Kalogerakis et al., 2017), voxel-based (Graham et al., 2018; Maturana and Scherer, 2015; Tchapmi et al., 2017; Wu et al., 2015) and point-based (Guo et al., 2021; Hu et al., 2020; Qi et al., 2017a; Wang et al., 2020; Zhao et al., 2021). Regarding ALS datasets, many methods are proposed based on deep learning, of which some are projection based (Hu and Yuan, 2016; Yang et al., 2017; Zhao et al., 2018) and some are point based (Huang et al., 2021; Li et al., 2020; Lin et al., 2021; Yousefhussien et al., 2018).

The deep learning algorithms mentioned above follow the typical training scheme for semantic segmentation where networks are trained fully supervised on pointwise labels over the entire training area. This scheme always requires a huge amount of precise pointwise labels. Unfortunately, the pointwise annotation for all points in datasets is difficult, especially for those covering large scale urban areas. This is

because points are unstructured and non-uniformly distributed. Manual pointwise annotation is very tedious and time-consuming, e.g. it takes over 2500 hours in total to manually label ALS point clouds of 2 km^2 with the density of 348 points/m^2 into 13 classes (Zolanvari et al., 2019). Therefore, it is necessary to investigate strategies to reduce the annotation efforts.

Various strategies are proposed to reduce manual annotation efforts on point clouds. Some methods utilize 2D data sources whose labels are much easier to obtain compared to 3D point clouds (Wang et al., 2019; Yang et al., 2020). However, producing precise annotation on 2D data is not cheap. The annotation cost can also be reduced by using low-cost noisy labels for the training of semantic segmentation of point clouds (Ye et al., 2021). Active learning is an alternative solution that aims to identify the most informative samples for model optimization and put manual annotation efforts only on those samples. Selected samples could be tiles that include complete scenes (Lin et al., 2020), super-points (Shi et al., 2021) or individual points (Kölle et al., 2021). Semi-supervised learning has also been investigated to alleviate annotation efforts. The idea is to assign semantic labels to a part of the points and models not only learn from the small set of labelled data but also exploit the potentials in the rest of the unlabelled data which take a larger proportion (Deng et al., 2022; Hu et al., 2021; Wang and Yao, 2021; Xu and Lee, 2020). Unsupervised learning is an alternative approach to addressing the problem. Pretrained models using contrastive learning loss are adapted for semantic segmentation of point clouds to reduce the annotation efforts. Contrastive losses are designed to force the model to learn from a large amount of unlabelled data (Hou et al., 2021; Xie et al., 2020). The underlying assumption is that spatial context and geometrical features of the same objects will not change with different rotations and perspectives. With the prior knowledge on point clouds acquired by unsupervised learning, the pretrained models only require a limited extra point annotation to achieve comparable results to those trained with fully labelled point clouds. The major limitation of the methods based on active, semi-supervised and unsupervised learning is that they require exact point annotations. However, it is difficult for non-experts to label 3D points and they need tutorials to learn how to use annotation tools to assign a sematic label to a point which will lead to more costs.

Inexact annotations on point clouds are easier to acquire because annotators do not need long tutorials to learn how to use annotation tools to label points and they only need to roughly outline objects or just recognize what kind of objects are in the scene. Although very few works attempt to train segmentation networks on those cheap labels

for point cloud tasks, this has been researched in many image related tasks. Various types of cheap labels are researched from bounding boxes (Dai et al., 2015; Song et al., 2019), scribbles (Lin et al., 2016), to image-level labels (Chang et al., 2020; Fan et al., 2020; Hou et al., 2018; Li et al., 2020; Oh et al., 2017; Stammes et al., 2020; Wei et al., 2017; Yu et al., 2019). From bounding boxes to image level labels, fewer and fewer localization cues are available to the training data and this is the main challenge when using inexact weak labels for semantic segmentation tasks.

Similar to weak supervision based on image-level labels, although the labels can be easily acquired, training semantic segmentation networks on weak subcloud labels is also very challenging. This is because pointwise labels are not available to train conventional semantic segmentation networks and only a label is given to a small region for the training data. Limited localization cues for objects make the training for semantic segmentation quite difficult. Strategies are required to establish how to correctly infer a semantic label for each point within a region without exact localization cues. Wei et al. (2020) propose a two-step framework to solve the problem, called multi-path region mining (MPRM). The first step is to generate pointwise pseudo labels from a classification network trained on weak labels and the second step is to take pointwise pseudo labels as the ground truth to train a segmentation network. They propose point class activation maps to provide the localization cues for target objects and four attention heads are proposed to obtain more precise pointwise pseudo labels for the training data. The MPRM mainly focuses on improving the pseudo label accuracy but does not take advantage of the overlapping weak labels which can be further exploited to provide extra spatial information. Also, they take all pseudo labels to train the segmentation network without considering the influence of the incorrect training labels.

Since accurate pointwise labels are difficult to acquire and weak labels on subclouds are more accessible, we investigate how to supervise the training for semantic segmentation of ALS point clouds by weak subcloud labels. In this chapter, we extend the MPRM proposed by Wei et al. (2020) and adapt it to ALS data. In order to obtain more accurate pseudo labels from the classification network, how to introduce more localization cues to the network without providing more manually annotated labels is investigated in our research. We propose an overlap region loss to exploit the semantic heterogeneity within a subcloud. Class labels of the overlap regions are implied through pairwise comparison between weak labels of overlapping subclouds and these implied labels provide semantic cues for different subregions of each

subcloud. Since elevation related features are important attributes to distinguish different target categories in ALS datasets and many methods take advantage of them, an elevation attention block is designed and embedded to MPRM in order to better identify objects in ALS training data. Compared to the classification network in MPRM (Wei et al., 2020), the proposed network can learn more representative class-specific features from ALS datasets. After preparing pseudo labels, instead of directly using all of them to train a segmentation network, we only select part of the pseudo labels as the ground truth for the training and the labels of the rest points are dynamically assigned during the training. Inspired by Khosla et al. (2020), we reveal the underlying correlations of class-specific features between different classes through a supervised contrastive loss to make full use of both labelled and unlabelled points. The main **contributions** of this work are listed as the following:

1) With weak subcloud labels covering large-area ALS point clouds, we make use of the overlapping subclouds and infer the label of the overlap from the label sets of the two overlapping subclouds. A loss function is designed to consider this inference and forces the pseudo label generation network to be aware of the semantic differences between a subregion and the entire subcloud, contributing to more accurate pseudo labels.
2) To fully exploit the characteristics of ALS datasets, an elevation attention block is introduced to the pseudo label generation network. It encodes elevation related features to higher-dimensional features and helps the pseudo label generation network to produce more class-specific features that are useful to identify different objects.
3) For the training with pseudo labels for ALS datasets, we adapt a supervised contrastive loss (Khosla et al., 2020) to reveal the underlying correlations of class-specific features between different classes.

In the rest of the chapter, we first review the related deep learning algorithms with both full and weak supervision for semantic segmentation of point clouds in Section 4.2. Next, our framework is explained and illustrated in Section 4.3. Section 4.4 shows our experimental results on three ALS datasets. Finally, Section 4.5 concludes this chapter.

4.2 Related work

4.2.1 Deep learning on point clouds

As point clouds are irregularly distributed, they cannot be directly taken as the input of 2D CNNs. Therefore, a branch of methods attempts to

convert 3D data into 2D representations, like producing images of 3D data from different views (Boulch et al., 2018; Kalogerakis et al., 2017) and these images can be fed into 2D CNNs for the 3D semantic segmentation tasks. The conversion from 3D to 2D is also applied to the semantic segmentation of ALS data, especially rasterizing features from the top view. Hu and Yuan (2016) create grid cells over XY plane and height related attributes within the cell are summarized and taken as the value of the corresponding pixel. In addition to height information, Yang et al. (2017) encode full-waveform and geometric features when generating 2D feature maps and feed those projected features to image based CNNs. Zhao et al. (2018) utilize multi-scale contextual images which further facilitate 2D CNNs to learn more representative features for ALS data. The limitation of these 2D CNN methods is that many pre-calculated features are required before the training and the training needs large memory to process the data.

Apart from projecting point clouds to 2D images, another approach to regularizing unordered point clouds is voxelization. The idea is to represent point clouds by regular 3D grids which can be processed by 3D convolutional networks (Maturana and Scherer, 2015; Tchapmi et al., 2017; Wu et al., 2015). However, the implementation of 3D CNNs can be inefficient when dealing with sparse points. To solve the problem, Graham et al. (2018) propose a sparse convolutional operation to speed up the convolutional calculation over voxelized data. The performance of the sparse submanifold convolutional networks (SSCNs) proposed by Graham et al. (2018) on the semantic segmentation of ALS data is researched by Schmohl and Sörgel (2019). The disadvantage of voxelization is that it causes 3D information loss. Recently, more and more researchers are attempting to directly take unevenly distributed points as the network input. PointNet proposed by Qi et al. (2017) is the first deep learning network that can take raw points as the input. PointNet learns geometrical features by a sequence of Multilayer Perceptron (MLP) layers. Based on PointNet, PointNet++ (Qi et al., 2017b) is composed of set abstraction modules that can progressively learn local geometrical features at different scales. RandLA-Net (Hu et al., 2020) has local feature aggregation modules to encode positional features and attentively pool them in a local neighbourhood. Unlike PointNet++ using farthest point sampling, Hu et al. (2020) prove that random sampling can also be an effective sampling strategy for representative feature extraction. WreathProdNet, proposed by Wang et al. (2020), is a hierarchical network where the wreath product of the group is utilized to express the symmetries of hierarchical structures. Inspired by the transformer utilized in natural language processing and image processing, Point

Cloud Transformer (Guo et al., 2021) and Point Transformer (Zhao et al., 2021) are proposed to capture better local geometrical features.

With the success of 2D convolutions in image related tasks, many researchers develop 3D convolutions to learn local geometrical features from point clouds. Different from 3D voxel based method, these 3D convolutions can directly take unstructured points as the input. For example, Kernel Point Convolutions (KPConv) (Thomas et al., 2019) are defined over continuous space where the weights of a point depending on the linear correlation between its position and its nearby kernel points' positions. The positions of kernel points are learnable, allowing the convolutions to learn more representative features from local structures. In addition to KPConv, Flex-Convolution (Groh et al., 2019), PointConv (Wu et al., 2019) and ConvPoint (Boulch, 2020) also define 3D convolutions over continuous space. In contrast, FKAConv (Boulch et al., 2020) is a convolutional operator defined over discrete space, which transforms irregularly distributed points to align with grid kernels.

There are also many deep learning based approaches proposed for the semantic segmentation of ALS datasets. Yousefhussien et al. (2018) allow the PointNet to learn from both XYZ coordinates and radiometric features captured from IR-R-G imagery. Winiwarter et al. (2019) introduce a batching framework that helps the PointNet++ to efficiently process large scale ALS point clouds. Li et al. (2020) design geometry-aware convolutions, construct a dense hierarchical network and propose an elevation-attention module to effectively encode the characteristics of ALS datasets. In order to capture representative features for ALS point clouds from local to global scales, Lin et al. (2021) first integrate features extracted from both 2D and 3D convolutions to extract local features. Then edge conditioned graph convolutions are applied to geometrical homogenous segments in order to exploit the contextual information at an object level. Finally, a spatial-channel attention is proposed to encode the dependencies at a global scale. These three blocks are embedded in a single network and can be trained end to end. Huang et al. (2021) propose a network GraNet to learn spatial dependencies at both local and global scales. Considering the point distribution of ALS data, they propose a local spatial discrepancy attention convolution module that encodes point distribution, orientation and elevation information. Then a global relation-aware attention module is proposed to further learn global structures and global dependencies in high-level features.

4.2.2 Weakly supervised semantic segmentation on 2D images

Before we review the weakly supervised methods on semantic segmentation of point clouds, we first review some weak supervision approaches proposed for image related tasks. Weak supervision for semantic segmentation on images has been researched in different kinds of supervisions like bounding boxes, scribbles and image level labels. Box level annotations provide rectangular masks to objects which are less precise but easier to obtain compared to pixel level masks (Dai et al., 2015; Song et al., 2019). Scribble annotations are easier to obtain compared to the bounding boxes as annotators only need to draw several lines for different objects (Lin et al., 2016).

Compared to annotations at the box and scribble levels, weak supervision with image-level annotations is the hardest task but it requires the least annotation efforts. Supervision by image-level labels mainly takes two steps. Firstly, pseudo masks of objects are derived from image-level annotations and then these masks are taken as the ground truth for the fully supervised training of semantic segmentation networks. To obtain pseudo masks of the objects, without providing any spatial information on target objects, taking advantage of class activation maps (CAM) is a common practice. CAM is a set of response maps for target categories derived from classification networks supervised by image level labels. However, CAM tends to pay attention to the most discriminative region of the target objects and fails to activate other object parts. To extend pseudo masks to less discriminative parts, a group of methods attempts to force the network to pay more attention to other parts of the objects. This can be achieved by hiding or erasing parts of target objects (Hou et al., 2018; Li et al., 2020; Stammes et al., 2020; Wei et al., 2017). Some approaches obtain more accurate pseudo masks by using features from different images (Chang et al., 2020; Fan et al., 2020). Other methods incorporate saliency maps in order to separate objects from their background (Oh et al., 2017; Yu et al., 2019). How to obtain accurate object masks based on image-level labels is the main issue these methods attempt to solve.

4.2.3 Deep learning on point clouds with fewer annotation efforts

Recently, some researchers have focussed on using limited accurate point annotation for the training of deep networks. Lin et al. (2020) propose an active learning strategy to select the most informative tiles to train PointNet++ for the semantic segmentation of ALS point clouds.

Instead of annotating tiles, Shi et al. (2021) propose an active learning strategy to progressively annotate the most informative segments which are clusters of points sharing similar geometrical features. With the intention to outsource the point annotation to the crowd, Kölle et al. (2021) iteratively select points which are easy for non-experts to interpret. Tao et al. (2020) introduce a one point per instance annotation strategy. Point clouds are first partitioned into homogenous segments and then one point annotation is assigned to the most representative segment per instance.

Apart from tiles and segments, supervision by sparse point annotation has also been investigated. Xu and Lee (2020) adopt a semi-supervised learning strategy to allow the network to learn from not only a small number of labelled points but also a large number of unlabelled points. To fully exploit unlabelled data, Xu and Lee (2020) introduce three constraints: inexact supervision ensures all categories at a block level are consistent with the pointwise labels in this block, self-supervision keeps the feature consistency when a point cloud is randomly rotated or flipped and a spatial and colour constraint smooths the network outputs. Wang and Yao (2021) also exploit the potentials of unlabelled points when limited accurate point labels are available. They introduce entropy regularization to penalize uncertain predictions, a consistency constraint for ensemble predictions and an online pseudo-labelling strategy to dynamically involve extra pointwise supervision. Deng et al. (2022) use sparse labels and train a graph convolutional network in a semi-supervised manner to generate foreground-background pointwise pseudo labels for training samples. Then the classification and segmentation losses are jointly minimized to enhance the model robustness to noises in pseudo labels. With sparsely labelled points, Hu et al. (2021) propose a point feature query network to spread sparse training signals to a larger spatial extent by encoding the most important features from local regions of annotated points. Zhang et al. (2021) first make a model to learn from unlabelled data by self-supervised training and then knowledge is transferred to the weakly supervised segmentation network to enrich the information learned from limited pointwise annotation. Moreover, during the weak supervision training, pseudo labels are also assigned to unlabelled points in order to expand the supervision information.

Some approaches take advantage of 2D labels which are easier to acquire. For example, Wang et al. (2019) build a graph convolutional network for semantic segmentation of point clouds that can achieve 2D and 3D joint optimization and only requires 2D supervision. Yang et al. (2020) supervise the training of semantic segmentation of ALS point clouds by 2D labels acquired from topographic maps. Inspired by weak

supervision with the image level annotation, Wei et al. (2020) first set up the baseline for weak supervision with scene and subcloud level annotations which are easier to acquire compared to point annotations. However, using subcloud level labels is more challenging than using point annotations because no spatial information is available for labels. Following CAM, a point cloud activation map (PCAM) is proposed to provide localization cues for target objects. In order to obtain better pseudo labels for the training data, four attention heads are introduced and a conditional random field is taken as the postprocessing to further refine pseudo labels. Pointwise pseudo labels are taken as the ground truth to train normal networks for semantic segmentation of point clouds. Our work follows Wei et al. (2020), investigating the potentials of subcloud labels for semantic segmentation of ALS data as they are the easiest annotation to acquire.

4.3 Method

We adopt and further develop the framework proposed by Wei et al. (2020) (Figure 4.1). There are two main steps, generating pointwise pseudo labels from a classification network (the upper part in Figure 4.1) and training a segmentation network based on pseudo labels (the lower part in Figure 4.1). In the first step, Wei et al. (2020) use weak labels on subclouds for the training area to train the classification network and pass the training data to the trained classification network in order to produce pointwise pseudo labels. We extend this step by introducing an overlap region loss and an elevation attention module. The overlap region loss is designed to exploit the weak label information between nearby subclouds and the elevation attention module aims to help the classification network learn more class-specific features from the ALS point clouds. In the second step, training data with pointwise pseudo labels are utilized to train a segmentation network which then gives predictions on unlabelled testing data. We further develop the segmentation network by applying a supervised contrastive learning loss to train the segmentation network, which allows the segmentation network (KPConv network (Thomas et al., 2019)) to learn better class-specific representations with inaccurate pseudo labels.

In this section, we first review the weak label settings and network design proposed by Wei et al. (2020) which are taken as our baseline. Then we describe the mechanism of the overlap region loss, the design of the elevation attention and how we apply the supervised contrastive loss.

4.3.1 Brief review on MPRM

4.3.1.1 Subcloud-level annotation

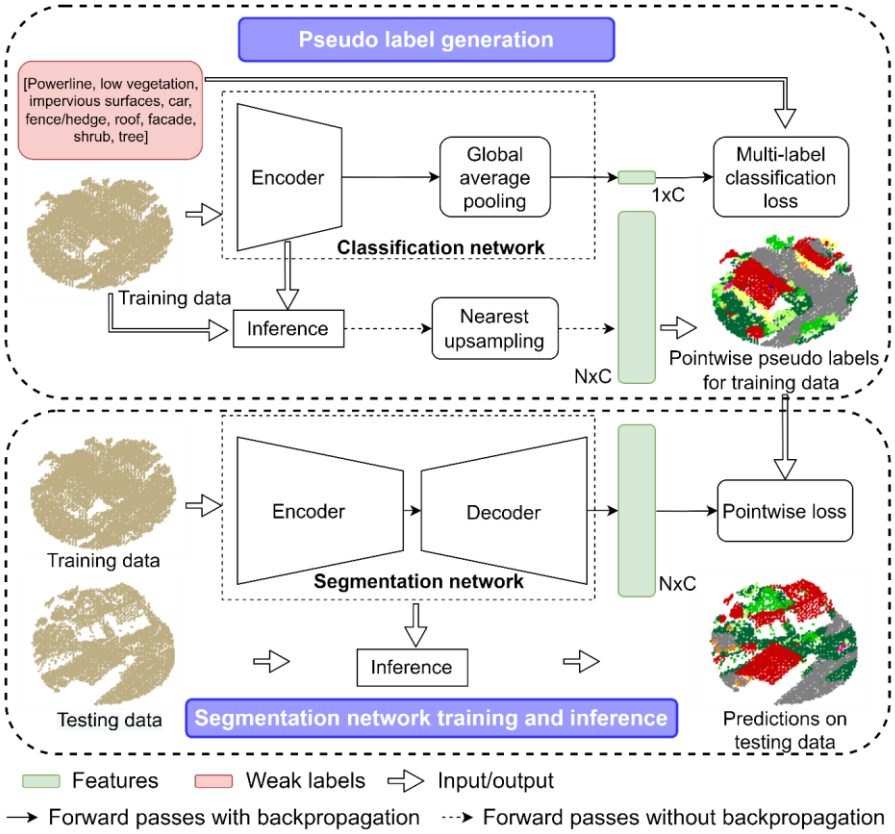


Figure 4.1 The workflow of weak supervision for the semantic segmentation of ALS data based on weak subcloud labels.

Wei et al. (2020) use subcloud-level labels for the weak supervision of 3D semantic segmentation. In MPRM, anchor points $A = \{a_1, a_i, \dots, a_{N_g}\}$ are uniformly placed in the space. For a_i, all neighbouring points within a radius of r_s are grouped as a subcloud g_i. Therefore, a large point cloud is separated into a set of subclouds $G = \{g_1, g_i, \dots, g_{N_g}\}$. A label vector $l_{si} \in R^{1\times C}$ is assigned to the subcloud g_i as the weak label. Elements in l_{si} are either 0 or 1. If $l_{si_c} = 1$, c_{th} category exists in g_i and c_{th} category is taken as a positive class. If $l_{si_c} = 0$, c_{th} category is not present in g_i and c_{th} category is taken as a negative class. When generating subclouds, anchor points A are uniformly placed along the XYZ axis and the distance between anchors is taken as the same as

the subcloud radius r_s. Therefore, the number of subclouds can be calculated as $\lceil l_x/r_s \rceil \times \lceil l_y/r_s \rceil \times \lceil l_z/r_s \rceil$, supposing no empty anchors are placed at empty space. Here, l_x, l_y, l_z represent the extents in the three dimensions of the training data. With this subcloud extraction strategy, the union of all subclouds covers the entire training area. Every point has a chance to be included in multiple subclouds and to be assigned different weak subcloud labels.

4.3.1.2 Point class activation map

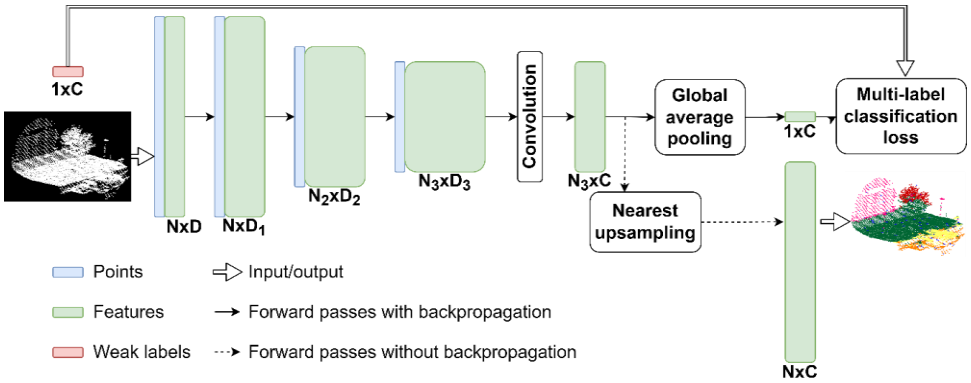


Figure 4.2 The structure of the base network to generate PCAM and pseudo labels. Here, the input sphere has ground, façade and vegetation points. The weak label can be written as a vector [1,0,0,1,1,0] which represents the existence of ground, roof, water, façade, vegetation and artwork in sequence.

The use of a class activation map (CAM) is common practice for many weak supervision tasks on semantic segmentation of images. It extracts class-specific object localization cues from classification networks. CAM is applied to point-based convolutional networks, called a point class activation map (PCAM) (Wei et al., 2020), in order to produce localization cues on 3D point clouds. The structure of the base network to generate PCAM, denoted as $f_{PCAM} \in R^{N_3 \times C}$, is shown in Figure 4.2. A subcloud g_i is passed through three convolutional layers to obtain the intermediate feature map $f^{[3]} \in R^{N_3 \times D_3}$. Then, the feature map is converted to the PCAM by a 1× 1 convolution layer which reduces the feature dimension to the number of classes. During the training, a global average pooling layer is applied to produce the 1D prediction vector $logits_i \in R^{1 \times C}$ and a multi-label classification loss is computed according to the weak label of the subcloud l_{si}. Here the binary cross entropy loss is taken as the multi-label classification loss. It not only encourages logits for positive classes to be higher but also

forces the logits for negative classes to be lower. The loss for g_i is written as the following:

$$Loss_i = BCE(logits_i, l_{si}) \tag{4.1}$$

$$BCE(logits_i, l_{si}) = -\sum_{c=1}^{C}[l_{si_c} \cdot logits_{i_c} + (1 - l_{si_c}) \cdot log(1 - logits_{i_c})]$$

4.3.1.3 Multi-Path Region Mining

In order to learn more discriminative features that can provide more reliable object localization cues, the intermediate feature map $f^{[3]} \in R^{N_3 \times D_3}$ is fed to three attention modules (Figure 4.3), namely spatial attention module, channel attention module and point-wise attention module. The idea is to make different attention modules focus on different perspectives of the network and then generate class-specific discriminative regions. The aggregation of them gives more reliable pseudo labels. As for the plain PCAM path, each attention module is followed by a 1× 1 convolution layer to obtain a corresponding PCAM. Then a global average pooling layer is applied to obtain a 1D prediction which is passed to a cross entropy loss function. During the training, the loss is summed up and back-propagated. All the paths are aggregated by taking the element-wise maximum and are then upsampled to produce pseudo labels.

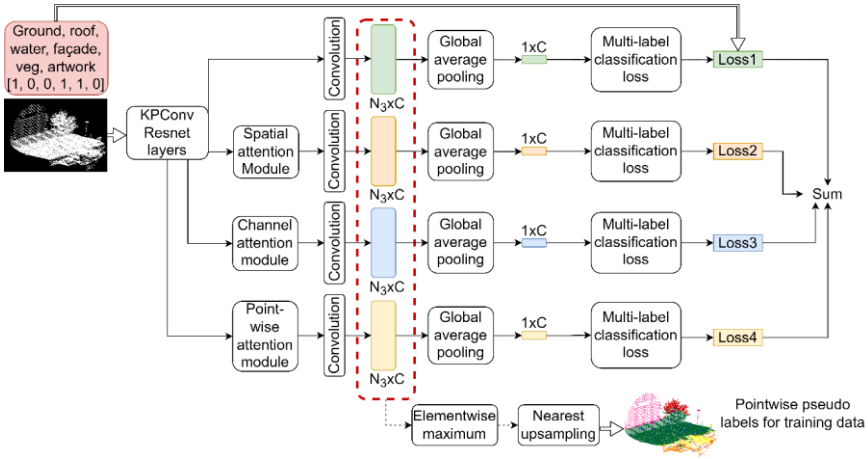


Figure 4.3 Pseudo label generation in MPRM (Wei et al., 2020).

4.3.2 Pseudo label generation

4.3.2.1 Reveal the relationship between subclouds

We follow the setup of MPRM to generate weak labels on subclouds. However, Wei et al. (2020) directly use a subcloud and its corresponding label vector for training, without considering the additional semantic information that can be inferred from two overlapping subclouds. Suppose we have two overlapping subclouds g_1 and g_2. g_1 has low vegetation, impervious surfaces and car points, while g_2 has low vegetation, impervious surfaces, etc. except car points (Figure 4.4). In this case, we can infer that the overlap region has no car, powerline, roof, façade, fence/hedge, shrub, tree points. This inference on negative classes can be utilized, since we use the binary cross entropy loss that tries to minimize the logits for negative classes.

To make use of weak labels of nearby subclouds, we compare subclouds by pairs (Figure 4.4) and then obtain pseudo weak labels for pairwise overlap regions. For a subcloud g_i, the overlap with a neighbouring subcloud g_j is denoted as g_{ij}. The label vector of g_{ij} is taken as $l_{sij} \in R^{1\times C}$, calculated as the following equation:

$$l_{sij} = l_{si} \circ l_{sj} \tag{4.2}$$

Where $\circ$ represents the Hadamard product and l_{sj} is the weak label for g_j.

Objects in the overlap g_{ij} can only be the positive classes l_{sij}, but not all positive classes in l_{sij} have to be present in the overlap. For example, in Figure 4.4, according to (4.2), low vegetation and impervious are positive classes in l_{sij} but, in reality, the overlap region does not contain low vegetation points. In contrast, negative classes in l_{sij} will never exist in the overlap region. Although some positive classes in inferred label vectors may not be present in the overlap, information on negative classes can also benefit the training cause we use the binary cross entropy that takes advantage of both positive and negative classes.

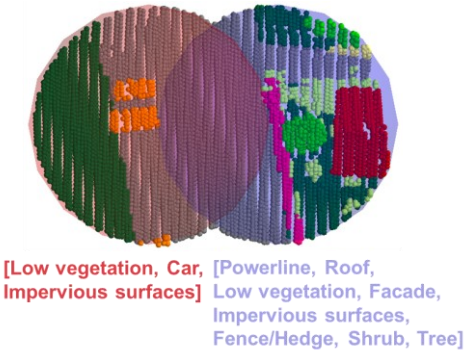


Figure 4.4 A sample of the overlap region.

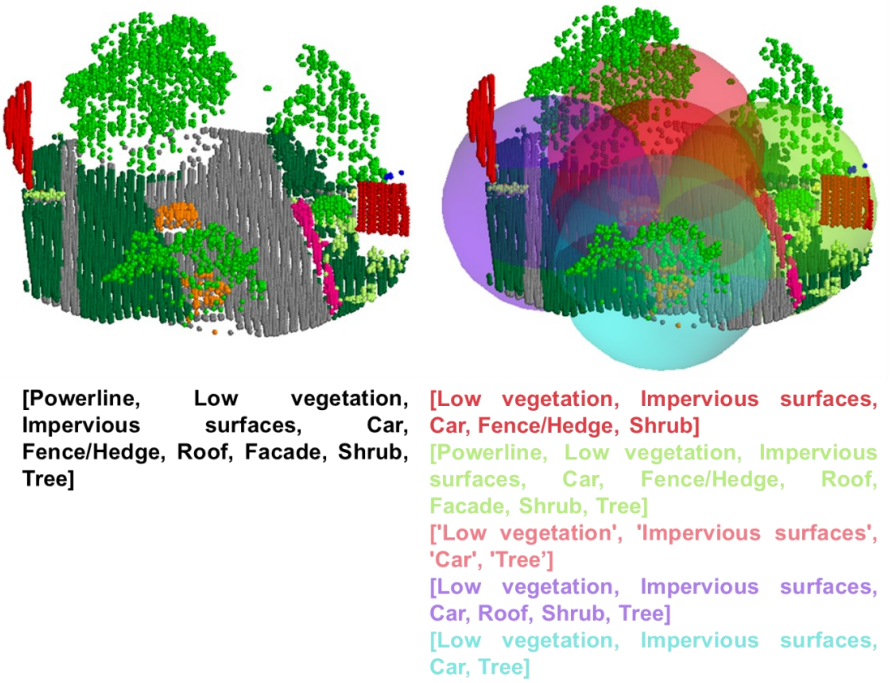


Figure 4.5 The relationship between the input sphere of the classification network and the subcloud for weak labels. The text below the point cloud is the weak labels provided to the input sphere. Each colour represents a weak label. The left one demonstrates the size of subcloud chosen by Wei et al. (2020), where $r_s = r_{in}$ (r_s =24m, r_{in}=24m). Only one weak label is assigned to the input sphere. The right one is the strategy used in our method, where $r_s < r_{in}$ (r_s=12m, r_{in}=24m) and five weak labels are available.

In our method, we add a term $Loss_{oi}$ to involve the pseudo weak labels for overlap regions.

$$Loss_{oi} = \sum_{j=1}^{M} BCE(logits_{ij}, l_{sij}) \tag{4.3}$$

Where $logits_{ij}$ is a feature vector computed by averaging all points in g_{ij} for each channel in PCAM. M is the number of subclouds that have an overlap with g_i. Therefore, the loss for g_i in our method is formulated as

$$Loss_i = BCE(logits_i, l_{si}) + \sum_{j=1}^{M} BCE(logits_{ij}, l_{sij}) \tag{4.4}$$

According to Thomas et al. (2019), the input of the KPConv is a spherical space whose radius is r_{in}. The subcloud is set as the same size as the input point cloud in MPRM (Wei et al., 2020) and therefore only one weak label vector is available for each input sphere. When we decrease the subcloud size ($r_s < r_{in}$), more subclouds can be included in the KPConv input (Figure 4.5) and the loss for the input point cloud can be written as:

$$Loss_{\ classification} = \sum_{i=1}^{N_s} Loss_i \tag{4.5}$$

where N_s is the number of subclouds within the input point cloud. N_s is 1 in MPRM (Wei et al., 2020).

4.3.2.2 Elevation attention

Elevation information has proven to be important to the semantic segmentation of ALS point clouds in many studies (Huang et al., 2021, Li et al. 2020). Therefore, it should be well-encoded to extract discriminative features for the network. In addition to taking elevation information as the input feature of the network, we design an elevation attention unit that encodes the elevation information to higher-dimensional features and helps the classification network to produce more class-specific features. The structure of our elevation attention unit is shown in Figure 4.6. The elevation attention is applied to the output feature of the last convolutional layer in the classification network, shown in Figure 4.7.

Here, we consider both the relative height in a local neighbourhood and the elevation in the original dataset, forming a height feature map $H \in R^{N\times 2}$. Then H is downsampled to the same size as $f^{[3]}$, denoted as H'. H' is transformed into two features maps f_{h1} and f_{h2} with the same dimension as $f^{[3]}$ by two 1× 1 convolution layers. The elevation attention EA is calculated by the matrix multiplication between the transpose of f_{h1} and f_{h2}, followed by a softmax function σ:

$$EA = \sigma({f_{h1}}^T \cdot f_{h2}) \tag{4.6}$$

Then a matrix multiplication is applied between EA and $f^{[3]}$ and the result is then multiplied by a learnable parameter α and element-wisely added to the intermediate feature $f^{[3]}$, shown in the following equation:

$$f^{[3]}{}_{EA} = \alpha \cdot EA \cdot f^{[3]} + f^{[3]} \tag{4.7}$$

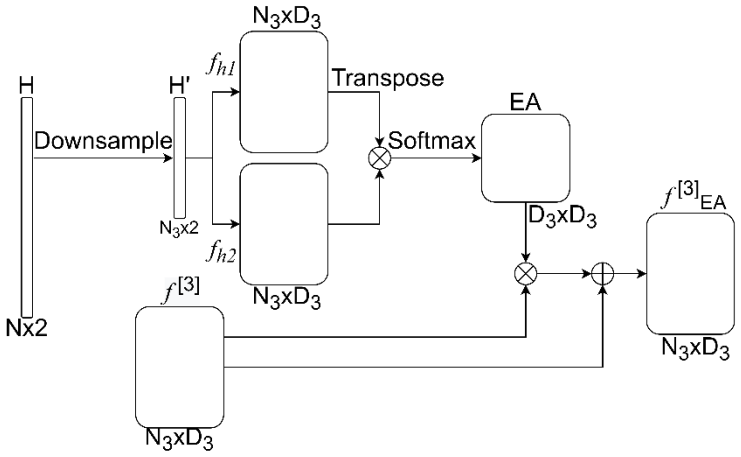


Figure 4.6 The structure of the elevation attention unit.

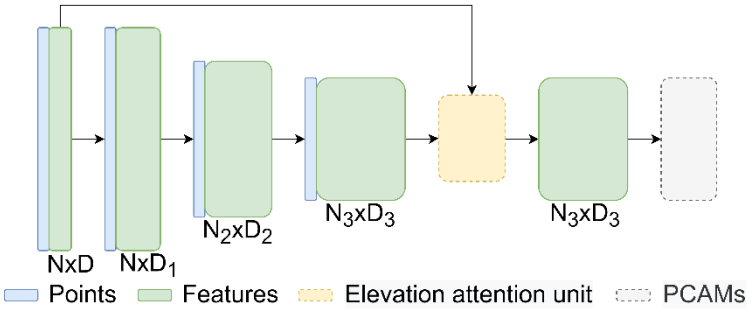


Figure 4.7 The position of the elevation unit.

4.3.2.3 Pseudo label refinement

Since each point may appear in multiple subclouds, it can correspond to several weak labels. These weak labels are helpful to mask out those classes that do not exist in the ground truth. For a point p, its new features $\hat{f}_{PCAM}(p) \in R^{1\times C}$ constrained by its corresponding weak labels can be defined as the following:

$$\hat{f}_{PCAM}(p) = softmax(f_{PCAM}(p)) \circ (l_{s1} \circ \ldots \circ l_{sk}) \tag{4.8}$$

Where k is the number of subclouds that include point p and $\circ$ is Hadamard product.

As our pointwise pseudo labels are inaccurate, we set a threshold t to select those confident predictions as the pseudo labels for the next training process and the rest are taken as the unlabelled points. Therefore, our pointwise label $\hat{y}$ for point p is obtained by the following equation:

$$\hat{y} = \begin{cases} None & , \quad \hat{f}_{PCAM}(p) < t \\ \underset{c \in C}{argmax} \left(\hat{f}^{c}{}_{PCAM}(p) \right), & \quad \hat{f}_{PCAM}(p) \geq t \end{cases} \tag{4.9}$$

4.3.3 Training with pseudo labels

4.3.3.1 Semantic segmentation network

As Wei et al. (2020) take the KPConv network as the segmentation network, in order to make a fair comparison to demonstrate the advantages of our proposed strategy, we follow Wei et al. (2020) to also take the KPConv network as the model for the semantic segmentation on the testing data. Its encoder consists of five convolutional layers and every layer has two convolutional blocks. The decoder contains four MLP layers. Suppose a set of points $P = \{p_1, p_i, \dots, p_n\}$ as the input of the network. Let $i \in I = \{1, \dots, N_p\}$ be the index of a point in P. The indices of labelled points can be written as $I_l \subseteq I$ and those of unlabelled points are taken as $I_u \subseteq I$. The numbers of points in the labelled and unlabelled sets are represented by N_{pl} and N_{pu} respectively. The weighted loss function based on labelled points is written as:

$$L_{ce} = \sum_{i \in I_l} w_{\hat{y}_i} \cdot CrossEntropy(\hat{y}_i, logits_i), \tag{4.10}$$

where $\quad w_{\hat{y}_i} = \frac{\ln\left(\frac{1}{\gamma_{\hat{y}_i}}\right)}{\sum_{\hat{y}_i=1}^{C} \ln\left(\frac{1}{\gamma_{\hat{y}_i}}\right)} \quad , \quad \gamma_{\hat{y}_i} = \frac{N_{\hat{y}_i}}{\sum_{\hat{y}_i=1}^{C} N_{\hat{y}_i}} \, , \, \hat{y}_i = \{1, \dots, C\}$

4.3.3.2 Supervised contrastive learning using pseudo labels

In order to the improve the representativeness of learned class-specific features, we introduce a contrastive learning strategy to allow the network to reveal the underlying correlation between different categories. In conventional self-supervised contrastive learning, for an anchor sample, positive samples are augmented samples derived from the same source sample while the negative samples are all other samples originating from different sources. Positive pairs are expected to have similar features while negative pairs are supposed to be different. Khosla et al. (2020) suggest that with the existence of labels

in supervised learning, positive samples can be those belonging to the same class. The supervised contrastive loss is proposed to pull together points in the same class in embedding space, while pushing apart samples from different classes. Since we do not have manually labelled pointwise labels, we use pointwise pseudo labels to define positive and negative samples for each point.

To take advantage of unlabelled points, instead of using fixed pseudo labels for training, we dynamically assign pseudo labels to those unlabelled points during the training if the network gives confident predictions on them.

$$\hat{y}_i = \underset{c \in C}{argmax}({f_i}^c), \qquad if\ softmax(f_i) \geq t, i \in I_u \tag{4.11}$$

where t is the same threshold mentioned in Section 4.3.2.3 and f_i is the feature map from the last layer of the segmentation network. Then, the labelled set is extended ${I'}_l = I_l \cup \{i\}$ and the unlabelled set is shrunk ${I'}_u = I_u \setminus \{i\}$.

For point $p_i\ (i \in {I'}_l)$, the indices of all other labelled points are represented as $V(i) = {I'}_l \setminus \{i\}$. The supervised contrastive loss (Khosla et al., 2020) for p_i in the extended labelled set ${I'}_l$ is defined as the following:

$$L_{cli} = \frac{-1}{|Z(i)|} \sum_{z \in Z(i)} log \frac{\exp\left(f_i \cdot f_z / \tau\right)}{\sum_{v \in K(i)} \exp\left(f_i \cdot f_v / \tau\right)} \tag{4.12}$$

Here, $Z(i) = \{z \in V(i) \colon \hat{y}_z = \hat{y}_i\}$ is the set of indices of all positives that share the same pseudo label with point p_i and its cardinality is denoted as $|Z(i)|$. The correlation between samples is calculated by a dot product between feature vectors followed by $\exp()$. f_i, f_z and f_v are pointwise features obtained by the last layer of the KPConv. τ is a temperature factor that controls the influence of the contrastive loss on gradients for back-propagation. A smaller τ means a larger gradient. Then supervised contrastive loss for all points with pseudo labels is computed as follows:

$$L_{cl} = \sum_{i \in {I'}_l} {w'}_{\hat{y}_i} \cdot L_{cli} \tag{4.13}$$

$${w'}_{\hat{y}_i} = \frac{1}{N_{\hat{y}_i}}, \qquad \hat{y}_i = \{1, \dots, C\} \tag{4.14}$$

To keep the balance between majority and minority categories, we assign weight ${w'}_{\hat{y}_i}$ to each pointwise loss according to its pseudo label. $N_{\hat{y}_i}$ is the number of points for category $\hat{y}_i$ in the labelled set. Finally, the L_{ce} and L_{cl} are summed up as the loss to train the segmentation network:

$$Loss_{segmentation} = L_{ce} + L_{cl} \quad (4.15)$$

4.4 Experiments

Experiments are carried out on ALS datasets to evaluate our weak supervision method based on subclouds. We first show our results on the ISPRS benchmark dataset (Niemeyer et al., 2014) and make comparisons to the baseline method (Wei et al., 2020) and other fully supervised methods. Experiments are performed to verify the effectiveness of our proposed method. Our weak subcloud supervision method is also tested on Rotterdam dataset (Lin et al., 2020) and DFC2019 dataset (Bosch et al., 2019) to demonstrate its pros and cons.

4.4.1 Experiments on ISPRS benchmark dataset

4.4.1.1 Dataset

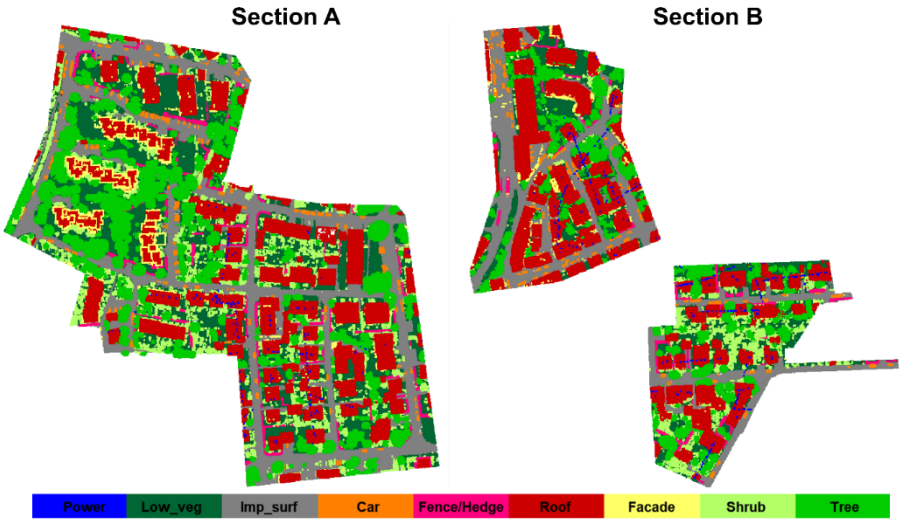


Figure 4.8 An overview of the ISPRS benchmark dataset. Section A is used for model training and Section B is used for model evaluation.

Our method is first tested on the ISPRS benchmark dataset (Niemeyer et al., 2014). Figure 4.8 provides an overview of the dataset which was captured by a Leica ALS50 system in August 2008, at Vaihingen, Germany with a mean flight height of 500m and a field of view of 45°. The point density is 4 points/m^2. Except for XYZ coordinates, the

dataset also provides pointwise intensity and number of returns. Nine classes are predefined in the dataset, namely powerline, low vegetation, impervious surface, car, fence/hedge, roof, façade, shrub, and tree. The point cloud is split into two parts, the training area with 753,876 points and the testing area with 411,722 points.

4.4.1.2 Accuracy assessment

We take the average F1 score (Avg. F1) and overall accuracy (OA) as our evaluation metrics to assess our proposed method. The overall accuracy is the proportion of correctly predicted points to the total number of points in the testing data. F1 score of each class is computed from precision and recall.

$$precision = TP \,/\, (TP + FP) \tag{4.16}$$

$$recall = TP \,/\, (TP + FN) \tag{4.17}$$

$$F1\ score = 2 \cdot (precision \cdot recall) / (precision + recall) \tag{4.18}$$

Where TP, FN and FP are true positives, false negatives and false positives receptively in a confusion matrix.

4.4.1.3 Preprocessing

To train the classification network which generates pseudo labels on the training data, we first generate subclouds with a radius of 6m. Following Wei et al. (2020), 7452 weak labels are generated for the training data with 753,876 points in total. The number of weak subcloud labels used for the training of the classification network is 1% of the number of points in the training data. For simplicity, we use '1% weak labels' to represent this weak labelling scheme. To demonstrate the superiority of our method over MPRM in terms of required annotation efforts and accuracy, we only use half of the weak labels ('0.5% weak labels') and mainly discuss the classification results under this scheme. How a smaller number of labels are selected is demonstrated in Section 4.4.1.6.2.

Following the data preprocessing utilized in KPConv, the ISPRS dataset is first subsampled by grid sampling with a grid size of 0.24m to mitigate the influence of the uneven spatial distribution of ALS data. Since both the classification network and segmentation network are built on KPConv, we apply the same strategy to generate input data for the model training. The radius of the input sphere is 24m. Apart from XYZ coordinates, we also use intensity, absolute height and normalized height as input features. A random rotation around the Z-axis is applied to each input sphere and random noises with a standard deviation of 4cm are added to XYZ coordinates to augment the point clouds without significant changes in object geometry.

4.4.1.4 Network implementation details

The encoder of the classification network has three convolutional layers and every layer has two convolutional blocks. The output dimensions for three convolutional layers are 64, 128 and 256 respectively. Their subsampling grid sizes are 0.24m, 0.48m and 0.96m in sequence and the corresponding convolution radii incrementally rise as 0.6m, 1.2m and 2.4m to have larger receptive fields for the latter layers. To find reliable pseudo labels, we set t as 0.2.

The encoder of the segmentation network consists of five convolutional layers and their output dimensions are 64, 128, 256, 512 and 1024 respectively. Their subsampling grid sizes are 0.24m, 0.48m, 0.96m, 1.92m and 3.84m in sequence and the corresponding convolution radii incrementally increase as 0.6m, 1.2m, 2.4m, 4.8m and 9.6m. For the supervised contrastive learning loss, since we have limited GPU memory, we cannot compare all points in a batch pair by pair. We randomly select 2000 points and perform pairwise comparisons, which means that cardinality of $K(i)$ is 1999. According to Khosla et al. (2020), we set the temperature factor τ as 0.1.

Our experiments rely on the PyTorch framework (Paszke et al., 2019) and are implemented on a Geforce RTX 2080 Ti GPU. Network weights are optimized through stochastic gradient descent (SGD). To train the classification network, we set 400 iterations for a single epoch. The learning rate is initially set as 0.01 and gradually decreases with a decay rate of 0.98 at every epoch. It takes 80 epochs to achieve network convergence. For the segmentation network, one epoch is set as 2000 iterations and the learning rate is set as 0.001 whose decay rate is 0.9 for every 5 epochs. We train the segmentation network for 25 epochs to achieve network convergence.

4.4.1.5 Classification results

Table 4.1 Confusion matrix of our method (0.5% weak labels) on the ISPRS benchmark testing data. Precision, recall and F1 score are shown for each category. The overall accuracy is 0.800 and the average F1 score is 0.631.

	Power	Low_veg	Imp_surf	Car	Fence/Hedge	Roof	Facade	Shrub	Tree
Power	249	5	0	0	0	137	43	1	165
Low_veg	0	88768	4215	42	110	1171	241	1705	2438
Imp_surf	0	24458	77067	77	5	209	52	85	33
Car	0	554	61	2501	81	111	0	387	13
Fence/Hedge	0	2441	21	41	411	464	17	3549	478
Roof	214	4376	0	3	0	100290	861	824	2480
Facade	12	1905	9	29	0	1590	5855	874	950
Shrub	0	8429	106	96	99	1356	518	8915	5299
Tree	1	2256	2	6	31	1035	192	5563	45140
Precison	0.415	0.899	0.756	0.674	0.055	0.920	0.522	0.359	0.832
Recall	0.523	0.666	0.946	0.895	0.558	0.943	0.753	0.407	0.792
F1	0.463	0.766	0.840	0.769	0.101	0.931	0.616	0.382	0.812

The performance of our proposed weak supervision is evaluated on the ISPRS benchmark. We show the confusion matrix of our classification results under the 0.5% weak labels scheme in Table 4.1. Classification results and errors are qualitatively shown in Figure 4.9 and Figure 4.10 respectively. Our method correctly predicts 80% of the testing points and the average F1 score is 0.631. As shown in Table 4.1, the F1 scores of six classes exceed 0.6, namely low vegetation, impervious surface, car, roof, façade and tree. For powerline points, although the F1 score is only 0.463 and they are likely to be misclassified as roof and tree points, it is still surprising to see these delicate structures can be recognized when only weak subcloud labels are given for the training dataset without any precise localization information. The precision of fence/hedge is quite low and the fence/hedge points are likely to be predicted as low vegetation and shrub points. This is because fence/hedge, vegetation and shrub are likely to coexist in the same subcloud and they share similar intensity values. The classification network is difficult to produce precise localization cues on fence/hedge points and fails to generate accurate pointwise pseudo labels for those points. The inaccurate pseudo labels mislead the training of the segmentation network which then consequently fails to correctly predict fence/hedge points.

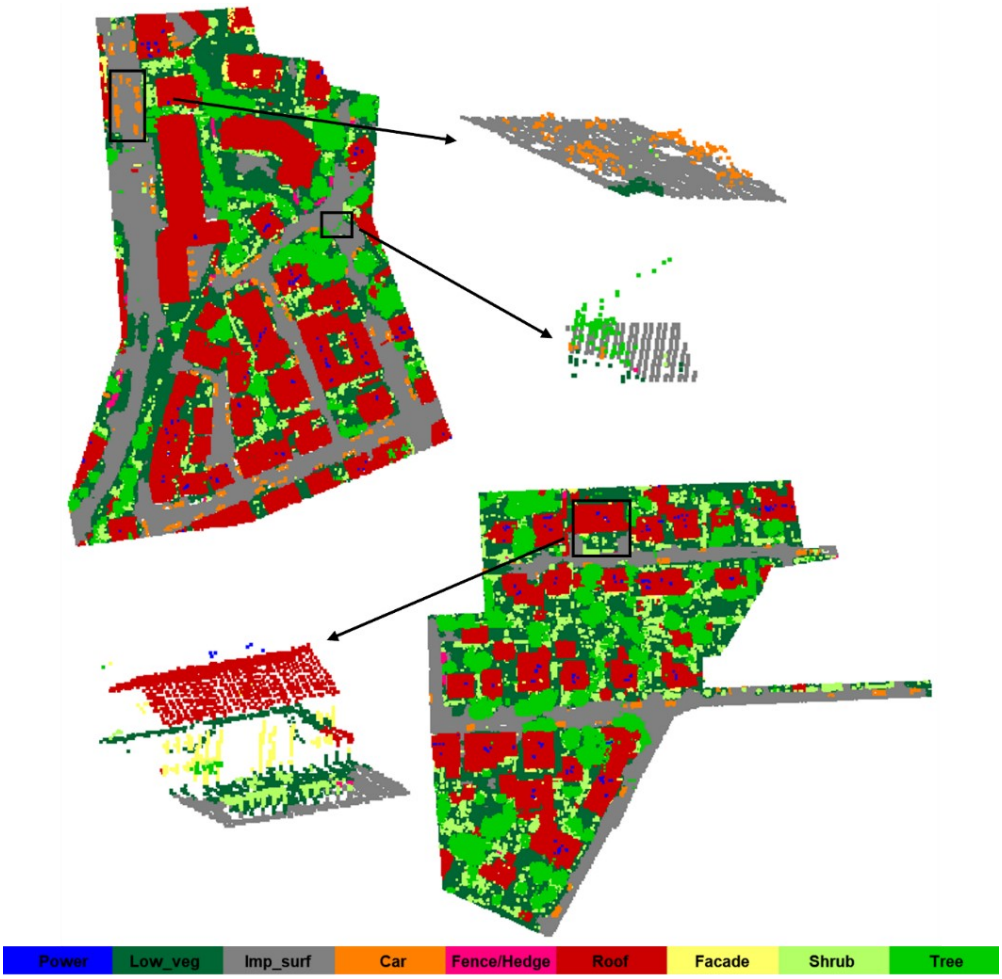


Figure 4.9 Classification results of our weak supervision on the ISPRS benchmark dataset (0.5% weak labels). A: Correctly predicted car points B: Wrongly predicted powerline points C: Correctly predicted powerline points over the roof.

We quantitatively compare our weak supervision method with some state-of-the-art full supervision methods on the ISPRS dataset in Table 4.2. All these methods are deep learning based, namely alsNet (Winiwarter et al., 2019), WhuY4 (Yang et al., 2018), KPConv (Thomas et al., 2019), RandLA-Net (Hu et al., 2020), GraNet (Huang et al., 2021), LGENet (Lin et al., 2021) and DAPnet (Chen et al., 2021). Compared to the best results (DAPnet), our segmentation results under 1% weak labels are 0.093 and 0.164 lower in terms of the overall accuracy and the average F1 score respectively. When reducing to half of the weak labels (0.5% weak labels), the accuracy for the prediction

on the testing data drops and the gap to DAPnet becomes 0.107 for the overall accuracy and 0.192 for the average F1 score. Compared to KPConv, the full supervision counterpart of our method, our results under the 1% weak labels scheme archive a comparable overall accuracy (0.814) and the average F1 score is 0.047 lower. Even if we reduce to half of the weak labels (0.5% weak labels), we still achieve 0.800 in the overall accuracy and 0.631 in the average F1 score. Although the F1 score is 0.075 lower than that of KPConv, the F1 scores in roof, façade and tree are comparable. The lower average F1 score can be explained by the poor performance on powerline and fence/hedge whose F1 scores are 0.286 and 0.212 lower than those obtained by KPConv respectively.

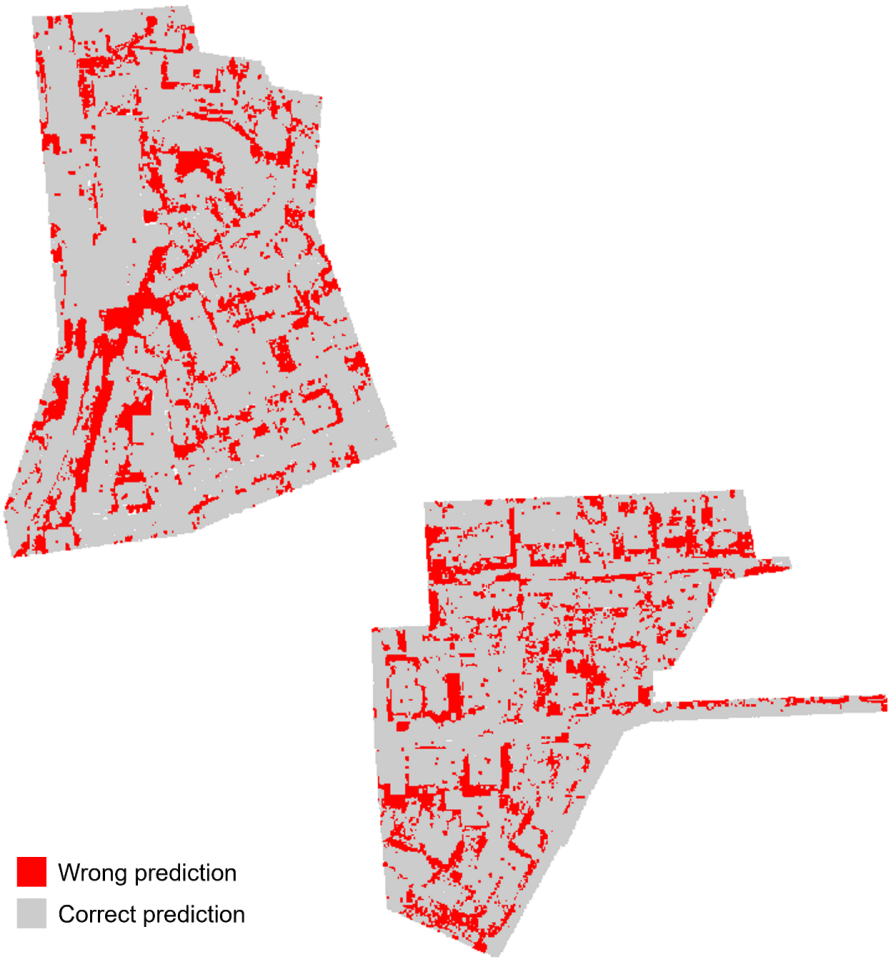


Figure 4.10 The error map of our weak supervision on the ISPRS benchmark dataset (0.5% weak labels).

Table 4.2 Quantitative comparisons between full supervision and weak supervision on the ISPRS benchmark dataset. Numbers in the first nine columns present the F1 scores for different categories. The overall accuracy (OA) and the average F1 score (Avg. F1) are listed in the last two columns.

Setting	Method	Power	Low_veg	Imp_surf	Car	Fence/Hedge	Roof	Facade	Shrub	Tree	Avg.F1	OA
Full supervision	aIsNet	0.701	0.805	0.902	0.457	0.076	0.931	0.473	0.347	0.745	0.604	0.806
	WhuY4	0.425	0.827	0.914	0.747	0.537	0.943	0.531	0.479	0.828	0.692	0.849
	KPConv	0.735	0.787	0.880	0.794	0.330	0.942	0.613	0.457	0.820	0.706	0.817
	RandLA-Net	0.690	0.801	0.914	0.731	0.362	0.937	0.584	0.458	0.827	0.701	0.831
	GraNet	0.677	0.827	0.917	0.809	0.511	0.945	0.620	0.499	0.820	0.736	0.845
	LGENet	0.765	0.821	0.918	0.800	0.406	0.938	0.647	0.499	0.836	0.737	0.845
	DAPnet	0.876	0.904	0.953	0.874	0.545	0.967	0.715	0.681	0.889	0.823	0.907
Weak supervision (subcloud)	MPRM (baseline 1%)	0.283	0.704	0.748	0.582	0.240	0.912	0.606	0.249	0.792	0.569	0.752
	Ours (0.5 %)	0.449	0.765	0.837	0.76	0.118	0.930	0.623	0.380	0.810	0.631	0.800
	Ours (1%)	0.457	0.774	0.874	0.752	0.289	0.931	0.612	0.428	0.820	0.659	0.814

Table 4.3. Quantitative results on ISPRS pseudo labels. Numbers in the first nine columns present the F1 scores for different categories. The overall accuracy (OA) and the average F1 score (Avg. F1) are listed in the last two columns.

Method	Power	Low_veg	Imp_surf	Car	Fence/Hedge	Roof	Facade	Shrub	Tree	Avg. F1	OA
MPRM (baseline 1%)	0.149	0.704	0.737	0.509	0.409	0.904	0.687	0.421	0.895	0.602	0.765
Ours (0.5%)	0.357	0.757	0.817	0.654	0.461	0.923	0.675	0.490	0.911	0.672	0.811
Ours (1%)	0.424	0.779	0.799	0.673	0.654	0.926	0.698	0.593	0.920	0.718	0.820

In Table 4.2, we also compare our proposed method to MPRM (Wei et al., 2020) which is taken as the baseline in this chapter. Following the strategy of producing weak labels in MPRM, 7452 weak labels are created for the ISPRS training data when the subcloud radius is taken as 6m. The baseline method only achieves 0.752 in the overall accuracy and 0.569 in the average F1 score. When using the same number of weak labels, our results (1% weak labels) achieve higher F1 scores in all categories. When reducing to half of the weak labels, our method still outperforms the baseline, especially on powerline, car and shrub. This can be explained by the more accurate pseudo labels

produced by our classification network shown in Table 4.3 and the contrastive learning loss applied to the segmentation network training. A more detailed explanation is shown in Section 4.4.1.6.

4.4.1.6 Experiments with different parameter settings

In the following paragraphs, we first implement experiments to find an optimal configuration for the classification network that produces accurate pseudo labels with a limited annotation budget. The different subcloud radii (Section 4.4.1.6.1), different extents of overlaps (Section 4.4.1.6.2) and the impact of elevation attention (Section 4.4.1.6.3) are tested for the classification network. Then we show how the performance of the segmentation network changes when using different confident levels of pseudo labels (Section 4.4.1.6.4) and how the supervised contrastive learning loss improves the results (Section 4.4.1.6.5).

4.4.1.6.1 The influence of subcloud radius

We conduct experiments to see how the performance of the classification network changes with different subcloud radii in order to find an optimal subcloud radius. We follow Wei et al. (2020) to generate subclouds with different radii. All experiments in this section use the baseline classification network. The overlap region loss and the elevation attention are not considered. Predictions on the testing data are produced by the original KPConv.

Table 4.4 Number of weak labels generated when using different subcloud radii. The ratio is calculated by the number of weak labels divided by the number of points in training data.

Subcloud radius (m)	6	12	24
Number of weak labels	7452	1626	325
Ratio	0.99%	0.22%	0.04%

Table 4.5 Pseudo label accuracy and testing data accuracy on the ISPRS dataset. The classification network is the same as MPRM (Wei et al., 2020). The KPConv is taken as the segmentation network (Thomas et al., 2019).

Subcloud radius (m)	Pseudo label accuracy		Testing data accuracy	
	Avg.F1	OA	Avg.F1	OA
6	0.602	0.765	0.569	0.752
12	0.462	0.673	0.476	0.684
24	0.318	0.579	0.311	0.592

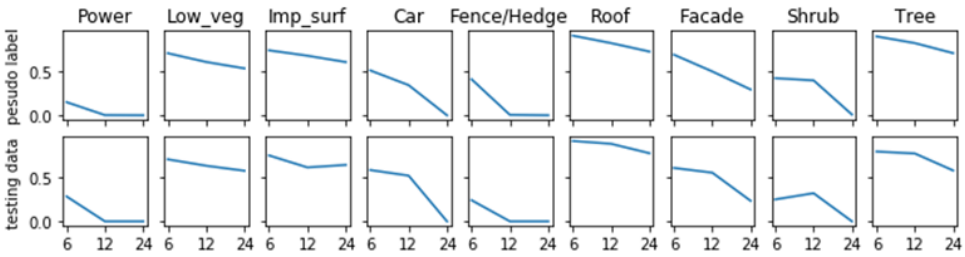


Figure 4.11 Pseudo label accuracy and testing data accuracy when using different radii of subcloud for weak supervision. In each plot, the x-axis represents subcloud radius (m) and the y-axis represents F1 score.

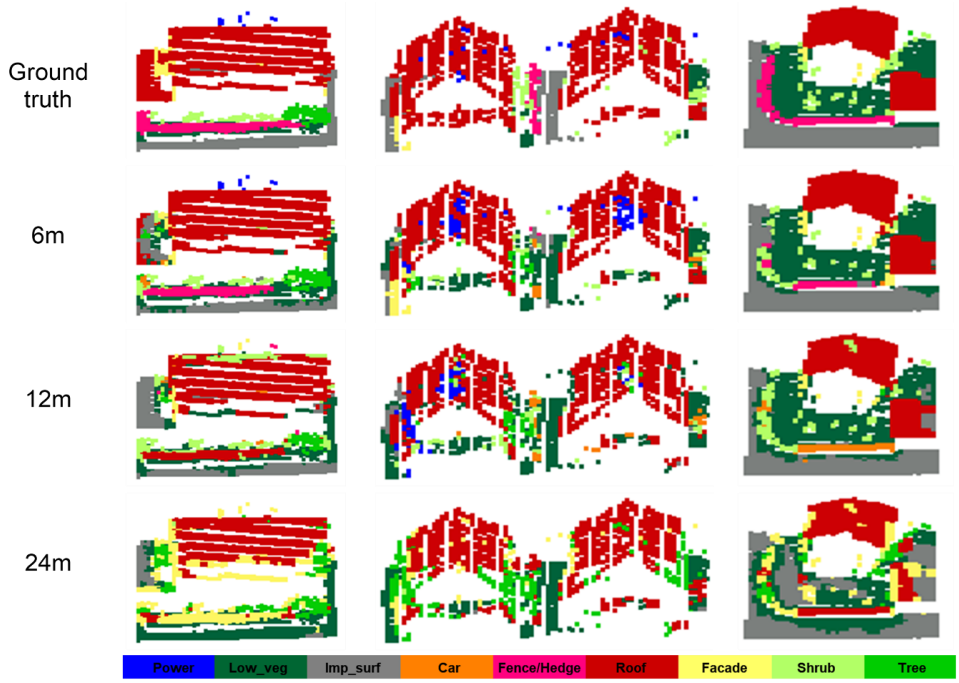


Figure 4.12 Examples of classification results on the Vaihingen dataset obtained from classification networks trained with different sizes of subclouds.

Our results suggest that the classification network prefers smaller subcloud radii. Table 4.4 demonstrates the number of weak labels produced under different scenarios. It can be seen that with a larger subcloud radius, fewer weak labels are available for the training of the model to generate pointwise pseudo labels in the training data. Pseudo label accuracies and classification results on testing data are

demonstrated in Table 4.5 and Figure 4.11. When the subcloud radius is 24m, only 325 subclouds are supposed to be manually labelled but these limited weak labels fail to train a model that can produce precise pseudo labels for small objects like powerline, car, fence/hedge (Figure 4.12). This is because the global pooling layer summarizes all the features to a vector and those small objects have very few contributions to the feature vector. Compared to those dominant classes, small objects are less likely to influence the loss and fail to make the training in favour of them. Consequently, the segmentation network trained on those unauthentic labels is not capable of identifying those classes. With decreasing subcloud radii, although more labelling efforts are required, each weak label represents fewer points and is likely to include fewer semantic classes. Points on small objects are likely to take a larger proportion of the subcloud and pose more impacts on the feature vector after the pooling layer. Therefore, these more exact weak labels allow the network to perceive small objects and give rise to better pseudo labels along with better predictions on testing point clouds. F1 scores in all categories are improved with the decrease of the subcloud size. It is possible to predict powerline, car, fence/hedge and shrub when the radius is 6m. Although 7452 are supposed to be given, this is still a small amount of labelling effort compared to the pointwise annotation on the whole training data. We also design an overlap region loss to further reduce half of the annotation without sacrificing the accuracy (Section 4.4.1.6.2). Therefore, we set the subcloud radius to 6m for the rest of our experiments on the ISPRS data.

4.4.1.6.2 The influence of overlap

As explained in Section 4.3.1.1, Wei et al. (2020) produce enough subclouds to make sure all the data for the training exist in at least two subclouds. However, do we need the whole region to be fully overlapped and can our overlap region loss help the weak supervision to save some labelling efforts without sacrificing the accuracy? Therefore, we directly drop some subclouds to reduce the proportion of the space covered by at least two subclouds. 60% means 60% of data space for the training is covered by at least two subclouds and 0% means there is no overlap between subclouds. In the following experiments, we first show how the results change with fewer weak labels. Then we analyse to what extent the overlap region loss can improve our results under different sampling strategies.

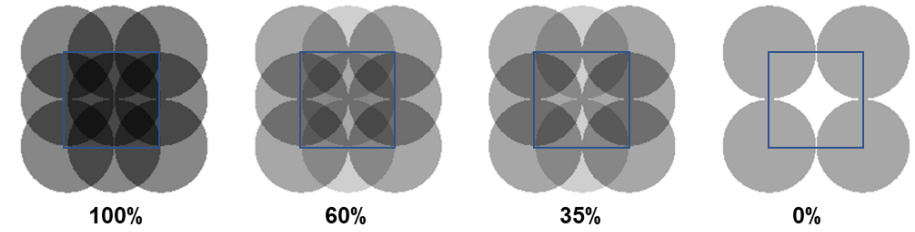


Figure 4.13 Examples of different sampling strategies to generate subclouds. Images are 2D projections of 3D objects. Blue squares represent the 3D space to be split. Grey circles represent 3D spheres. The number of spheres in each example is 27, 15, 14 and 8 from left to right. The darker grey means the region is covered by more spheres in the 3D space. For instance, in the example of 100%, there are three balls aligned along the edge that is represented by the upper corner of the blue square, while in the case of 60%, only 2 balls are available to that edge. The difference between the examples of 60% and 35% is that 60% has one more sphere in the middle.

Table 4.6 Number of labelled subclouds with different percentages of the overlap region. The ratio is calculated by the number of weak labels divided by the number of points in training data.

Percentage of the overlap region	0%	35%	60%	100%
Number of weak labels	953	2762	3624	7452
Ratio	0.13%	0.37%	0.48%	0.99%

The following section shows results when using different levels of overlap. Figure 4.13 demonstrates how subclouds are selected under different sampling strategies and Table 4.6 lists the corresponding number of weak labels produced. Note that the overlap between the 3D spheres is much smaller than visible from their projections in Figure 4.13. In general, more weak labels give rise to better performance of the classification network. Table 4.7 shows the accuracies of the pseudo labels and corresponding predictions on testing data respectively. When the classification network does not consider the overlap region loss, with more overlapped regions, the average F1 score and the overall accuracy increase in the aspect of pseudo labels. More weak labels avoid the overfitting of the network and allow the network to learn representative features from the whole training data. When it comes to the F1 scores for different categories (blue lines in the first row of Figure 4.14), most of the classes have better F1 scores, except fence/hedge (Figure 4.15). This is because fence/hedge points always coexist with low vegetation and shrub points and these three categories share similar intensity values (as mentioned in Section 4.4.1.5). More overlaps mean more subclouds, where fence/hedge

points coexist with low vegetation or shrub points. The enhanced learning from this coexistence confuses the network to correctly distinguish fence/hedge points from low vegetation or shrub points. Therefore, the pseudo labels of fence/hedge points are more likely to be taken as low vegetation or shrub when the overlap is set to 100%.

For predictions on testing data, the average F1 score and the overall accuracy increase when the percentage of the overlap region is larger for the classification network. F1 scores for different classes are plotted as blue lines in the second row of Figure 4.14. Due to the improvement in pseudo label accuracy, the F1 scores for most of the classes increase. Although the F1 score in the pseudo label of fence/hedge is higher when the overlap region is set to 60%, this high value does not bring advantages to the accuracy in prediction on testing data. This can be explained by the false positives for fence/hedge labels in the pseudo label. More shrub and low vegetation points are labelled as fence/hedge points for the training and therefore more shrub points are predicted as fence/hedge, leading to a larger number of false positives. This amount of misclassified shrub points has little influence on the F1 score in shrub because of the large number of shrub points but brings about the decrease in the F1 score in fence/hedge.

Table 4.7 Quantitative results on pseudo labels and predictions on testing data under different overlap schemes.

	Pseudo labels				Predictions on testing data			
	No overlap region loss		With overlap region loss		No overlap region loss		With overlap region loss	
Overlap	Avg.F1	OA	Avg.F1	OA	Avg.F1	OA	Avg.F1	OA
0%	0.439	0.659	0.439	0.659	0.447	0.691	0.447	0.691
35%	0.557	0.726	0.598	0.761	0.520	0.706	0.554	0.750
60%	0.591	0.741	0.629	0.782	0.546	0.739	0.584	0.760
100%	0.607	0.763	0.639	0.793	0.578	0.755	0.632	0.803

When the overlap region loss is considered, pseudo labels and corresponding predictions on testing data become more accurate with larger overlapped regions. How the F1 scores for different classes change with larger overlapped regions are plotted as yellow lines in Figure 4.14. By comparing yellow (overlap loss) and blue (no overlap loss) lines in the first row of Figure 4.14, it can be seen that the overlap region loss improves the pseudo label in most of the classes and contributes to a better average F1 score and overall accuracy. Since we infer weak labels from overlapping subclouds, the inferred labels correspond to subregions within a subcloud. If a small object exists in the subregion, it will give larger contributions to the pooling operation on the subregion and will be perceived by the classification network.

Apart from positive categories in weak labels, negative categories inferred from the overlaps also benefit the training, since the BCE loss penalizes high values on negative categories. This helps the network learn more discriminative features to distinguish different classes. In terms of the F1 scores for each class, the improvement is more significant for small objects like powerlines and cars, and this suggests the effectiveness of our overlap region loss in providing better localization cues. The loss fails in fence/hedge when the overlap is set to 60% and 100% due to its coexistence with shrub and low vegetation.

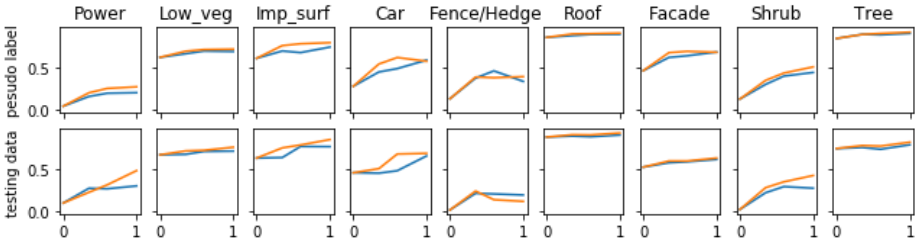


Figure 4.14 Quantitative results on pseudo labels over the training data and predictions on testing data when using different percentages of the overlap region for weak supervision. The first nine columns show the F1 score for different categories. Blue lines represent experiments that do not use the overlap region loss. Yellow lines represent experiments that use the overlap region loss. The x-axis for each figure is the percentage of the overlap region and the y-axis represents the F1 score.

When it comes to predictions on testing data (the second row of Figure 4.14), most categories take the advantage of better pseudo labels and achieve better accuracies, except fence/hedge. It can be concluded that our overlap region loss plays a greater role in obtaining a better average F1 score when the overlap is larger (Table 4.7). This is mainly related to better classification results on those difficult categories like powerline, car and shrub. By comparing the prediction on testing data using 100% overlap without overlap region loss and 60% with overlap region loss (Table 4.7), it can be found that the latter scheme achieves better accuracy in terms of the average F1 score and the overall accuracy. Referring to Table 4.6, this suggests that simply using an overlap region loss can reduce half of the required manual annotation effort. We take the overlap as 60% for the following experiments (presented in Table 4.8-4.10).

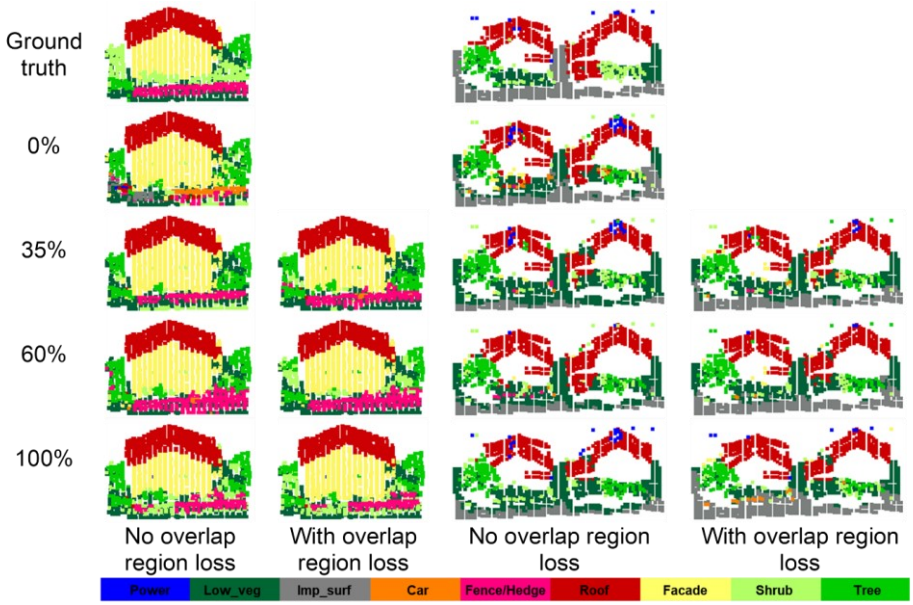


Figure 4.15 Examples of classification results on the Vaihingen dataset obtained from classification networks trained under different overlap schemes.

4.4.1.6.3 Effectiveness of elevation attention

Table 4.8 Quantitative results on pseudo labels and predictions on testing data with and without elevation attention (0.5% weak labels scheme).

Method	Pseudo labels		Predictions on testing data	
	Avg.F1	OA	Avg.F1	OA
Without elevation attention	0.629	0.782	0.584	0.760
With elevation attention	0.672	0.811	0.600	0.788

Quantitative results of the pseudo labels produced by models with and without elevation attention and the corresponding predictions on the testing data are shown in Table 4.8 and Figure 4.16. The classification network in this section involves the overlap region loss but the segmentation network is still the original KPConv. For pseudo labels, the average F1 score and the overall accuracy rise 0.043 and 0.029 respectively when using the elevation attention. The elevation attention increases the F1 scores for 8 out of 9 categories. It corrects car points that are misclassified as roof points. Also, it can correct impervious surface points on top of the buildings to roof points (Figure 4.17). The attention module reweights pointwise output features from the backbone according to elevation related features and makes the

features from different heads more class-specific. However, it confuses the network to label façade points near roof boundaries as roof and therefore reduces the precision of the façade. When comparing the predictions on testing data (in the second row of Figure 4.16), the misclassification between roof and façade points in pseudo labels does not negatively influence the F1 score in the façade. However, the F1 score in fence/hedge and shrub decreases. This is because the attention elevation fails to raise the recall of fence/hedge in pseudo labels and more shrub points are labelled as fence/hedge, leading to confusion between these classes in the testing data.

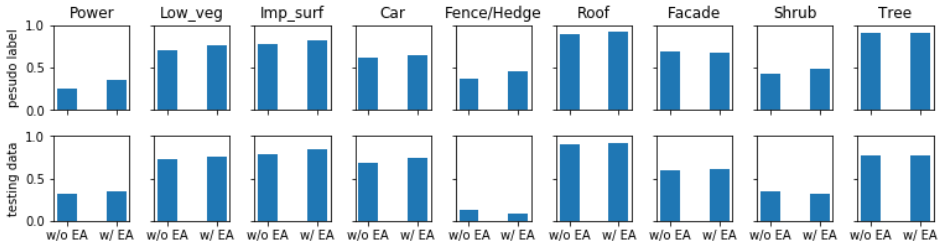


Figure 4.16 Quantitative results on pseudo labels and predictions on testing data when the classification network without (w/o) and with (w/) elevation attention (EA). In each plot, the x-axis represents whether the elevation attention is used and the y-axis represents F1 score.

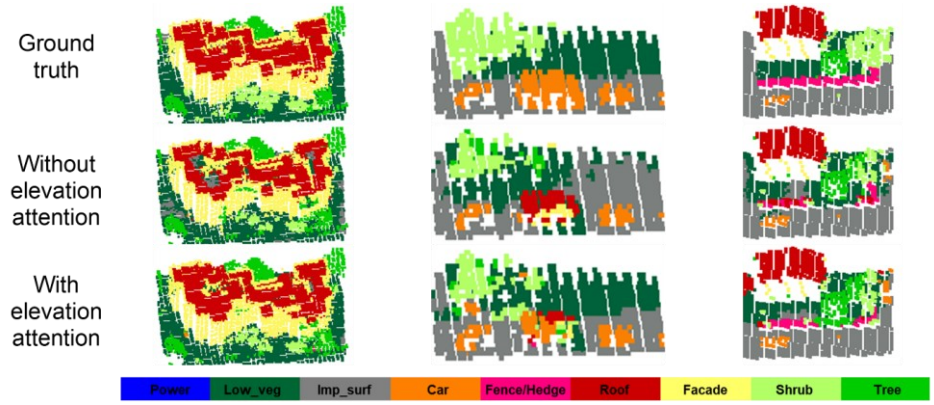


Figure 4.17 Examples of classification results on the Vaihingen dataset obtained from classification networks trained with and without elevation attention (0.5% weak labels scheme)

4.4.1.6.4 Influence of threshold t

All classification results on testing data are obtained by models trained with all pseudo labels. However, not all generated pseudo labels are correct. Therefore, we set a threshold t to select only confident pseudo

labels for the training. How the threshold influences the selected pseudo label accuracy is demonstrated in Figure 4.18. The selected pseudo labels are more accurate with a larger threshold. Figure 4.19 shows how the predictions on testing data change with different threshold values. It can be seen that larger threshold values have very little influence on the overall accuracy but fail to raise the average F1 score. More confident pseudo labels give rise to the slightly increased F1 score in roof and tree and the success in these dominant categories ensure the overall accuracy. The decreased average F1 score is a result of the failure in predicting powerline points (Figure 4.19). The reason for this failure is that a stricter threshold leads to a smaller proportion of powerline points in the whole training data. When the threshold is set to 0.3 and 0.4, the number of powerline points in pseudo labels is too small to allow the KPConv to learn representative features for themselves. Simply dropping unconfident labels by a threshold cannot improve the predictions on the testing data. Therefore, how to make use of those dropped points should be explored.

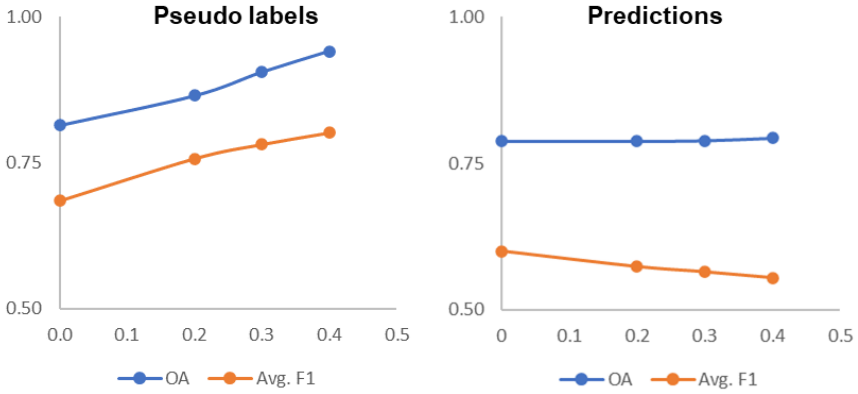


Figure 4.18 Quantitative results on selected pseudo labels and predictions on test data with different values of the threshold for the ISPRS benchmark dataset.

Table 4.9 Classification results on the ISPRS testing data when using different thresholds (t) to select confident points for training.

Threshold	0	0.2	0.3	0.4
Avg.F1	0.6	0.574	0.565	0.555
OA	0.788	0.788	0.789	0.793

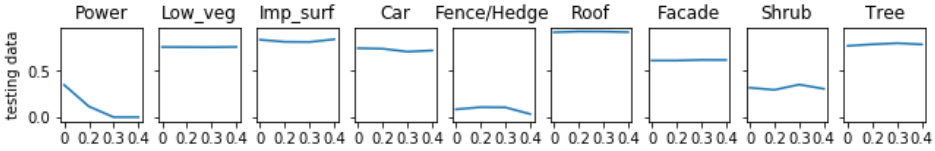


Figure 4.19 Classification results on the ISPRS testing data when using different thresholds ($\boldsymbol{t}$) to select confident points for training. In each plot, the x-axis represents the threshold value and the y-axis represents F1 score.

4.4.1.6.5 Effectiveness of supervised contrastive learning

Table 4.10 and Figure 4.20 demonstrate the classification results on testing data with supervised contrastive learning loss under different thresholds (t). Compared to Figure 4.19, the segmentation network trained with supervised contrastive learning loss produces better results than its counterpart in terms of the average F1 score. When the threshold is set to 0, there are no uncertain points and therefore no pseudo labels are dynamically generated during the training. It can be seen that simply using contrastive loss can help the network to learn more representative features for difficult categories whose F1 scores are less than 0.6, like powerline, fence/hedge and shrub. The supervised contrastive learning loss contributes to the most significant improvement when the threshold is set to 0.2. The average F1 score increases by 0.057 and the overall accuracy rises by 0.012. The explanation for this is twofold. Firstly, with the contrastive loss, the model can learn better representative features by pulling together points in the same category and pulling apart points in the different categories in the embedding space. Secondly, during the training, since the segmentation network gradually converges with a decreasing loss, the model can progressively produce more and more accurate pointwise pseudo labels for uncertain points. Compared to the training of the classification network, the training of the segmentation network can perceive small objects through the existing exact pseudo labels. Then those small objects are likely to be correctly labelled when they appear in the unlabelled points. Points with corrected pseudo labels make contributions to the loss and then allow the segmentation network to perceive them. However, when the threshold value is set to 0.3 and 0.4, our contrastive loss fails to make the training on pseudo labels in favour of powerline points due to the small number of powerline points.

Table 4.10 Classification results on ISPRS testing data when using different thresholds ($\boldsymbol{t}$) to select confident points for training.

Threshold	0	0.2	0.3	0.4
Avg.F1	0.624	0.631	0.581	0.563

OA	0.788	0.800	0.791	0.796

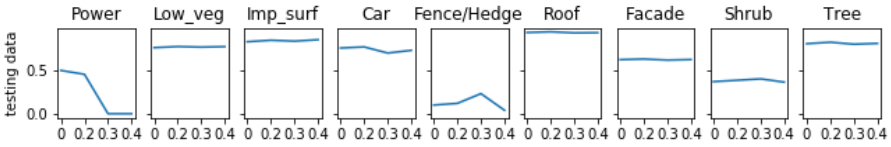


Figure 4.20 Classification results on the ISPRS testing data when using different thresholds ($\boldsymbol{t}$) to select confident points for training. The results are obtained by models trained with supervised contrastive learning loss. In each plot, the x-axis represents the threshold value and the y-axis represents F1 score.

Model sizes and runtime of different classification and segmentation models are listed in Table 4.11 and Table 4.12 respectively. In terms of the classification networks, compared to the MPRM, our proposed overlap region loss does not lead to more trainable parameters but it does increase the processing time for each input sphere. The extra time is a result of loss calculation and backpropagation on subregions within each input sphere. Adding an elevation attention module directly enlarges the network and slightly rises the running time. The running time is also influenced by the number of weak labels available for the training. When we keep the network the same and only double the number of weak labels (third and fourth rows in Table 4.11), the running time of our proposed method increases by 144%. The reason for this long processing time is that more weak labels on overlapping regions are inferred by pairwise comparison between nearby subclouds when more weak labels on subclouds are available over the training data. More overlapping regions give rise to a longer training time for each input sphere. Since there is no calculation of the overlap region loss in MPRM, more weak labels will not influence its training time per input sphere. For the segmentation network, the supervised contrastive loss does not increase the model complexity. The longer training time is caused by the update of the pseudo labels on unlabelled points, the calculation of the supervised contrastive loss and the backpropagation. Compared to the MPRM, although our models take a longer time to run, our results are significantly improved as shown in Table 4.10.

Table 4.11 Classification network size and runtime comparison among different methods.

MPRM	Overlap_Region_Loss	Elevation_Attention	Weak_Labels	Params (M)	Training time/input sphere (s) (Vaihingen dataset)
×			0.50%	4.5	0.05
×	×		0.50%	4.5	0.08
×	×	×	0.50%	5.4	0.09
×	×	×	1.00%	5.4	0.22

Table 4.12 Segmentation network size and runtime comparison among different methods.

	Params (M)	Training time/input sphere (s) (Vaihingen dataset)
KPConv	4.1	0.018
KPConv with supervised contrastive loss	4.1	0.031

4.4.2 Experiments on Rotterdam dataset

4.4.2.1 Dataset

The Rotterdam dataset is built on the third version of Actueel Hoogtebestand Nederland (AHN) dataset which provides ALS point clouds with a coverage of more than half of the Netherlands. The point cloud we used in Rotterdam is obtained by an IGI LM6800 system with a 60° field of view. The data acquisition was carried out on 4th December 2016 with a mean strip overlap of 30% and was designed to produce point clouds at a density of 60 points/m^2. The Rotterdam dataset is located in the city centre with a coverage of 2 × 2 km^2. It is characterized by high-rise buildings surrounded by trees and river channels with bridges. There are six classes predefined including ground, roof, water, façade, vegetation and artwork. The artwork class includes bridges and constructions related to water management. Figure 4.21 gives an overview of the dataset and how training and testing areas are split.

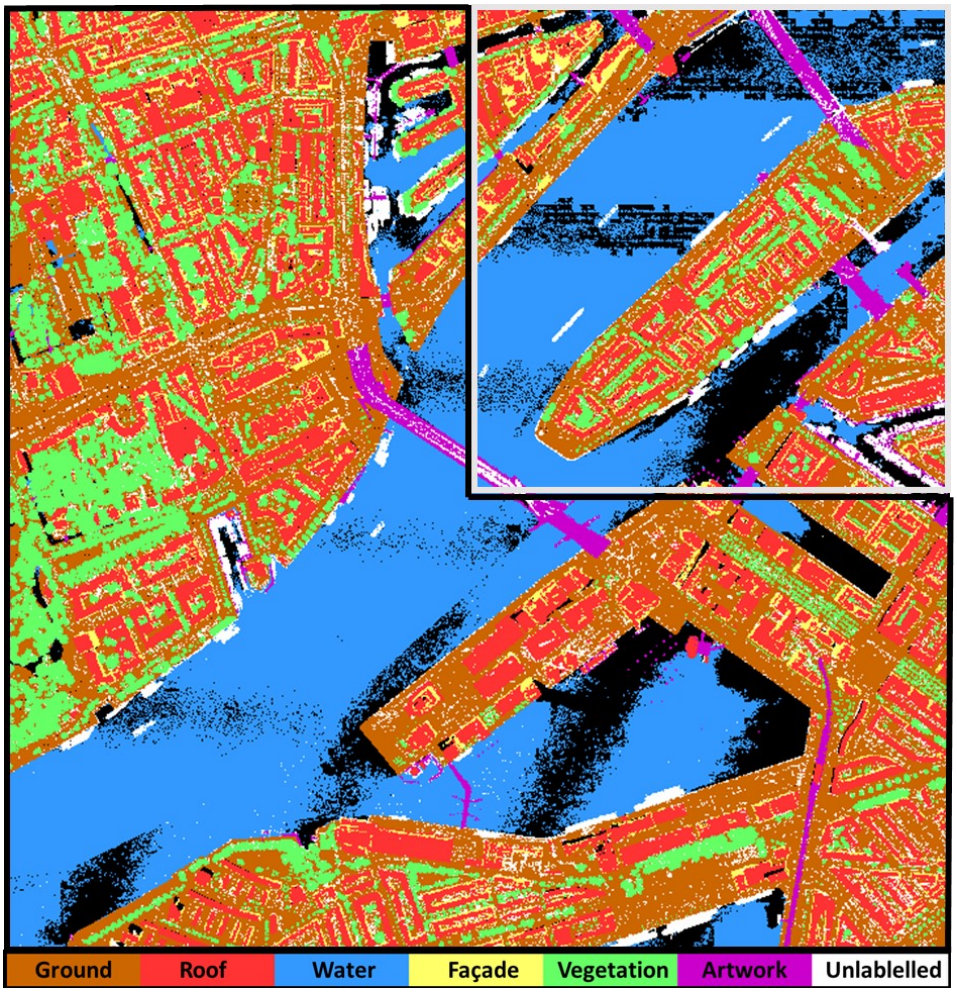


Figure 4.21 An overview of the Rotterdam dataset. The training area is in the black box and the testing area is in the grey box.

4.4.2.2 Preprocessing

Since the Rotterdam dataset only has six classes and small objects like cars and powerlines are not considered in the dataset, we take the subcloud radius as 24m which is the same size as the input sphere to the network. With 100% overlap, we generate 10604 weak labels for the area of 3 km^2 containing about 45 million points and we simplify this scheme as '0.02% weak labels'. On average, each subcloud contains 2.78 categories and it takes us about 1s to label one category. Therefore, it takes about 3s on average to manually label a subcloud and 8.8 hours is estimated to provide weak labels for the whole training dataset. However, for the same data, it takes us about 130 hours to

give pointwise labels. We also tried 60% overlap which only needs 6259 weak labels, denoted as '0.01% weak labels'.

Similar to the ISPRS dataset, we first downsample the Rotterdam dataset with a grid size of 0.24m and we use the same input data generation strategy for both classification and segmentation networks. We take the same input sphere radius (24m) and the same data augmentation strategy as the experiments on the ISPRS dataset. Absolute height and normalized height are taken as the input features. The network structures of the classification and segmentation networks are the same as those used for the ISPRS benchmark dataset.

To train classification and segmentation networks, we use the same parameters as the training on the ISPRS benchmark dataset, except epochs. We train the classification network for 100 epochs and the segmentation network for 30 epochs to achieve convergence.

4.4.2.3 Classification results

Table 4.13 Quantitative comparisons between full supervision and weak supervision on the Rotterdam testing data. Numbers in the first six columns present the F1 scores for different categories. The overall accuracy (OA) and the average F1 score (Avg. F1) are listed in the last two columns.

Setting	Method	Ground	Roof	Water	Façade	Veg	Artwork	Avg. F1	OA
Full supervision	KPConv	0.967	0.930	0.992	0.874	0.982	0.769	0.919	0.950
Weak supervision (subcloud)	MPRM (baseline 0.02% weak labels)	0.914	0.726	0.966	0.734	0.616	0.210	0.694	0.813
	Ours (0.02% weak labels)	0.947	0.916	0.971	0.806	0.962	0.632	0.872	0.925

Table 4.13 quantitatively compares the classification results on the Rotterdam testing data obtained by full supervision and weak supervision. Compared to the full supervision where models trained with pointwise ground truth labels, predictions obtained by our method using 0.02% weak labels are 0.025 lower in the overall accuracy and the gap in the average F1 score is 0.047. The main errors are with façade and artwork points. Although the gaps between our results and the full supervision predictions are not more than 0.02 for ground, roof, water and vegetation, only using a small number of weak labels for the training is still difficult to obtain precise localization cues for categories like façade and artwork. In terms of weak supervision by subcloud

labels, our method achieves much better accuracy compared to the baseline method (Wei et al., 2020). Table 4.14 and Table 4.15 more details on the advantages of our method.

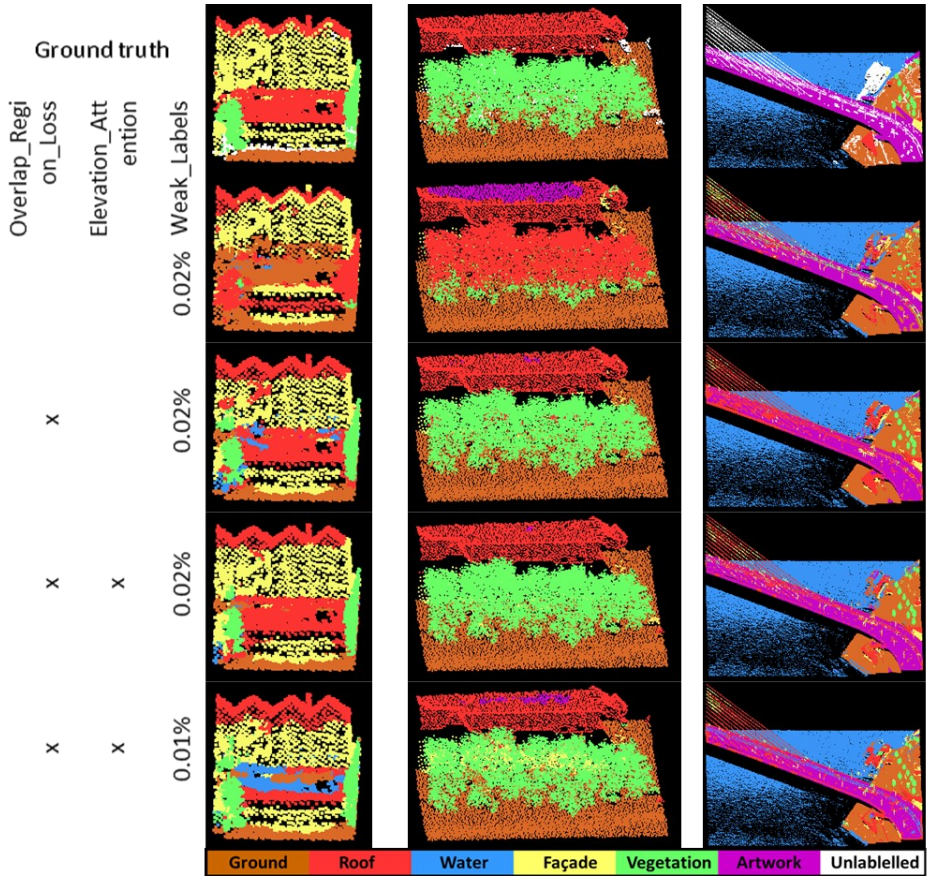


Figure 4.22 Qualitative results on pseudo labels for the training data of the Rotterdam data.

Table 4.14 quantitatively presents the pseudo labels generated by different training strategies. Compared to the baseline method (Wei et al., 2020) under the 0.02% weak labels scheme, our overlap region loss significantly improves all the F1 scores for all classes, contributing to the increases in the average F1 score and the overall accuracy by 0.191 and 0.192 respectively. By comparing the third row to the second row, it can be seen that using elevation attention does not make improvements in the overall accuracy but slightly increases the average F1 score. The elevation attention corrects some roof points on the bridge (Figure 4.22). When using 0.01% weak labels, pseudo labels produced by our proposed method achieve 0.773 in the average F1

score and 0.862 in the overall accuracy, which are better than the MPRM using 0.02% weak labels.

Table 4.14 Quantitative results on pseudo labels on the Rotterdam dataset obtained by different strategies for pseudo label generation. The baseline is the MPRM (Wei et al., 2020).

Overlap_Region_Loss	Elevation_Attention	Weak_Labels	Ground	Roof	Water	Façade	Veg	Artwork	Avg. F1	OA
		0.02%	0.872	0.643	0.853	0.546	0.481	0.432	0.638	0.733
×		0.02%	0.958	0.894	0.865	0.72	0.967	0.571	0.829	0.925
×	×	0.02%	0.951	0.914	0.87	0.697	0.969	0.598	0.833	0.925
×	×	0.01%	0.896	0.888	0.802	0.638	0.858	0.555	0.773	0.862

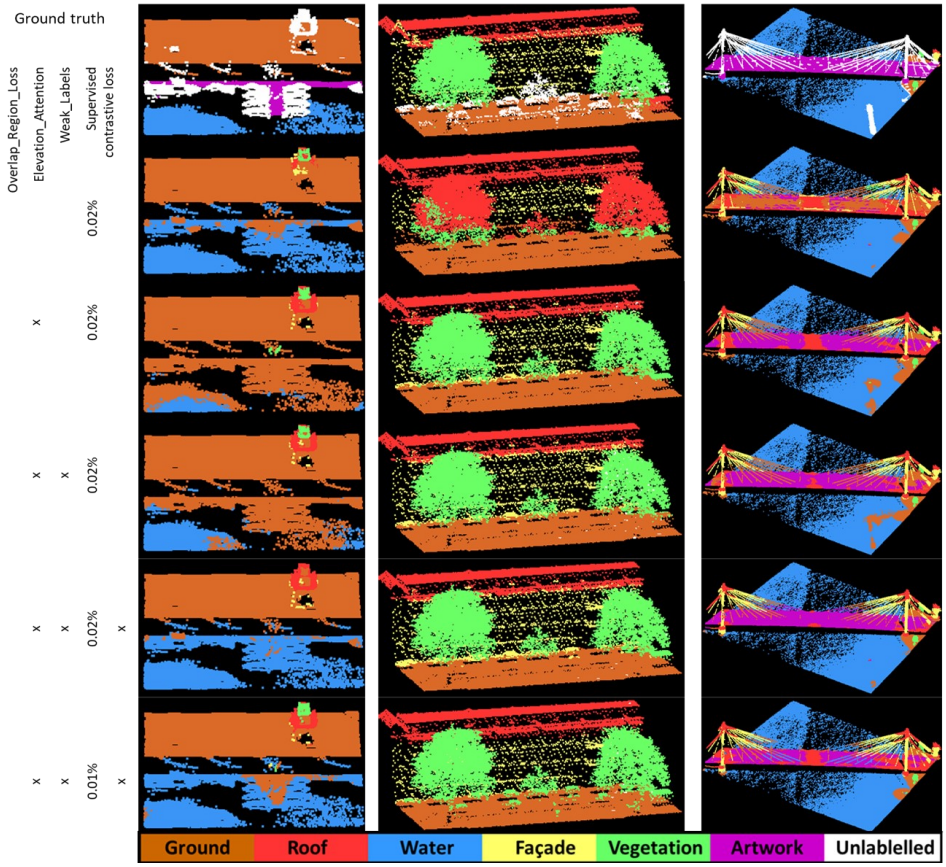


Figure 4.23 Qualitative results on pseudo labels for the training data.

The predictions on the testing data trained based on the pseudo labels mentioned above are quantitatively shown in Table 4.15 and qualitative shown in Figure 4.23. Due to more accurate pseudo labels obtained by the model using the overlap region loss, results on the testing data rise by 0.161 and 0.102 in terms of the average F1 score and the overall accuracy respectively, compared to the baseline method. Using elevation attention in the classification network, the average F1 score rises from 0.855 to 0.866 and the overall accuracy increases from 0.915 to 0.924. With the supervised contrastive learning loss, our average F1 score is further increased by 0.006. We also test our method using 0.01% weak labels. Reducing half of the weak labels for our method leads to drops in the average F1 score and the overall accuracy, which are 0.034 and 0.023 respectively. However, compared to the baseline method, less weak labels still achieve better accuracy.

Table 4.15 Predictions on the Rotterdam testing data obtained by different training strategies. The first three rows show the predictions of the KPConv networks fully supervised trained by different pseudo labels. The fourth and fifth rows present the predictions of the segmentation network trained with the supervised contrastive loss.

Classification_Network			Segmentation_Network	Ground	Roof	Water	Façade	Veg	Artwork	Avg. F1	OA
Overlap_Region_Loss	Elevation_Attention	Weak_Labels	Supervised_contrastive_loss								
		0.02%		0.914	0.726	0.966	0.734	0.616	0.210	0.694	0.813
×		0.02%		0.934	0.903	0.970	0.808	0.967	0.547	0.855	0.915
×	×	0.02%		0.944	0.914	0.978	0.821	0.964	0.575	0.866	0.924
×	×	0.02%	×	0.947	0.916	0.971	0.806	0.962	0.632	0.872	0.925
×	×	0.01%	×	0.927	0.895	0.951	0.781	0.933	0.543	0.838	0.902

4.4.3 Experiments on DFC dataset

4.4.3.1 Dataset

DFC 2019 is an ALS dataset published by the IEEE Geoscience and Remote Sensing Society (GRSS) in 2019 (Bosch et al., 2019). The dataset was captured in two cities namely Omaha, Nebraska and Jacksonville, Florida in the United States, covering about 100 km^2 and consisting of 200 million points. The aggregate nominal pulse spacing is 0.8m for the point cloud acquisition process. Except for XYZ coordinates, pointwise intensity and return number are available. The dataset is labelled into five semantic categories: ground, high vegetation, building, water and bridge deck. Following the data split in

Wen et al. (2020), 100 and 10 tiles are taken as the training and the testing data respectively.

4.4.3.2 Preprocessing

The subcloud radius for the DFC2019 dataset is taken as 24m which is the same size as the input sphere to the network. 92304 weak labels are given under the 100% overlap scheme. Since the training dataset has about 75 million points, we simplify this scheme as '0.12% weak labels'. We also test with 60% overlap where only 48387 weak labels, denoted as '0.06% weak labels'. We first downsample the DFC2019 dataset with a grid size of 0.48m and we use the same input data generation strategy for both classification and segmentation networks. Intensity, absolute height and normalized height are taken as the input features. The network structures of the classification and segmentation networks are the same as those used for the ISPRS benchmark dataset. To train classification and segmentation networks, we use the same parameters as the training on the ISPRS benchmark dataset, except epochs. We train the classification network for 160 epochs and the segmentation network for 40 epochs to achieve convergence.

4.4.3.3 Classification results

Table 4.16 Quantitative comparisons between full supervision and weak supervision on the DFC testing data. Numbers in the first five columns present the F1 scores for different categories. The overall accuracy (OA) and the average F1 score (Avg. F1) are listed in the last two columns.

Setting	Method	Ground	High Vege-tation	Build-ing	Water	Bridge Deck	Avg. F1	OA
Full supervision	KPConv	0.991	0.975	0.893	0.434	0.694	0.797	0.978
Weak supervision (subcloud)	MPRM (baseline 0.12% weak labels)	0.964	0.876	0.630	0.000	0.000	0.494	0.922
	Ours (0.12% weak labels)	0.978	0.948	0.757	0.000	0.000	0.537	0.953

Classification results on the DFC testing data are quantitively shown in Table 4.16. Our method achieves 0.537 in the average F1 score and 0.953 in the overall accuracy, which is better than the baseline method. However, there is still a large gap between our weak supervision method and the fully supervised KPConv, especially for the average F1 score. Weak supervision methods fail to correctly predict water and bridge deck points although our methods can produce some pseudo labels for these two categories (Figure 4.24). The main reason is their geometrical disparities in training and testing data. Our method can only recognize large water areas instead of sparse water points near

pools (Figure 4.24). However, most of the water points are distributed along the riverbank (Figure 4.25). As a result of inaccurate labels for small bridges crossing over channels, weak supervision methods cannot identify bridge decks in the testing data.

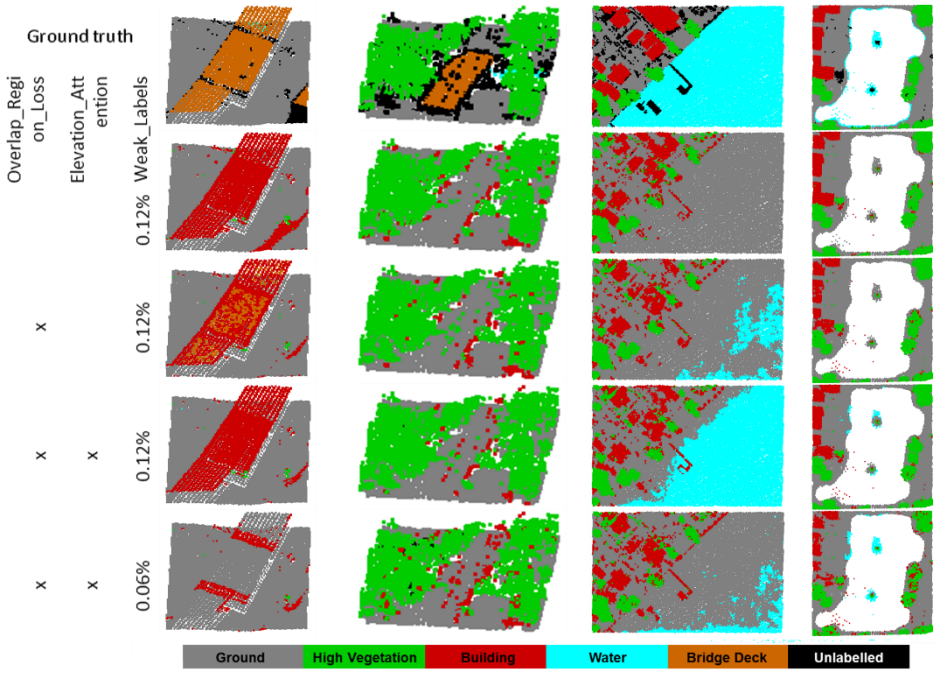


Figure 4.24 Qualitative results on pseudo labels for the training data.

Table 4.17 Quantitative results on pseudo labels on the DFC2019 dataset obtained by different strategies for pseudo label generation. MPRM is the baseline method following Wei et al. (2020).

Overlap_Region_Loss	Elevation_Attention	Weak_Labels	Ground	High Vegetation	Building	Water	Bridge Deck	Avg. F1	OA
		0.12%	0.927	0.886	0.594	0.002	0.000	0.482	0.864
×		0.12%	0.933	0.920	0.604	0.279	0.203	0.588	0.877
×	×	0.12%	0.933	0.919	0.629	0.551	0.000	0.607	0.879
×	×	0.06%	0.924	0.899	0.556	0.354	0.000	0.547	0.863

The pseudo label results on the DFC2019 dataset are quantitively listed in Table 4.17. The baseline method can barely give predictions on water and bridge deck points. Compared to the baseline method, our overlap region loss contributes to improvements in the average F1 score and the overall accuracy and makes a part of water and bridge

deck points detectable. Buildings in DFC2019 datasets are large in size and some of their roofs are large flat planes. Therefore, building points on roofs are likely to be predicted as ground and water which are also characterized as large plat planes. The elevation attention corrects those misclassified building points. However, it fails to recognize bridge deck points because bridge decks in the training data not only include the bridges crossing over channels but also contain large elevated roads which are large flat planes higher than the ground and have similar geometry to buildings (Figure 4.25). As bridge deck is not a majority class in the dataset, those points are misclassified as buildings when using elevation attention. When using 0.06% weak labels, our method achieves 0.547 in the average F1 score and 0.863 in the overall accuracy, which is comparable to the baseline method using 0.12% weak labels.

Table 4.18 Predictions on the DFC2019 testing dataset obtained by different training strategies. The first three rows show the predictions of the KPConv networks fully supervised trained by different pseudo labels. The fourth and fifth rows list the predictions of the segmentation network trained with the supervised contrastive loss.

Classification_Network			Segmentation_Network	Ground	High Vege-tation	Build-ing	Water	Bridge Deck	Avg. F1	OA
Overlap_Region_Loss	Elevation_Attention	Weak_Labels	Supervised contrastive loss							
		0.12%		0.964	0.876	0.630	0.000	0.000	0.494	0.922
×		0.12%		0.973	0.933	0.678	0.000	0.000	0.517	0.942
×	×	0.12%		0.978	0.934	0.736	0.000	0.000	0.530	0.950
×	×	0.12%	×	0.978	0.948	0.757	0.000	0.000	0.537	0.953
×	×	0.06%	×	0.966	0.943	0.608	0.000	0.000	0.503	0.935

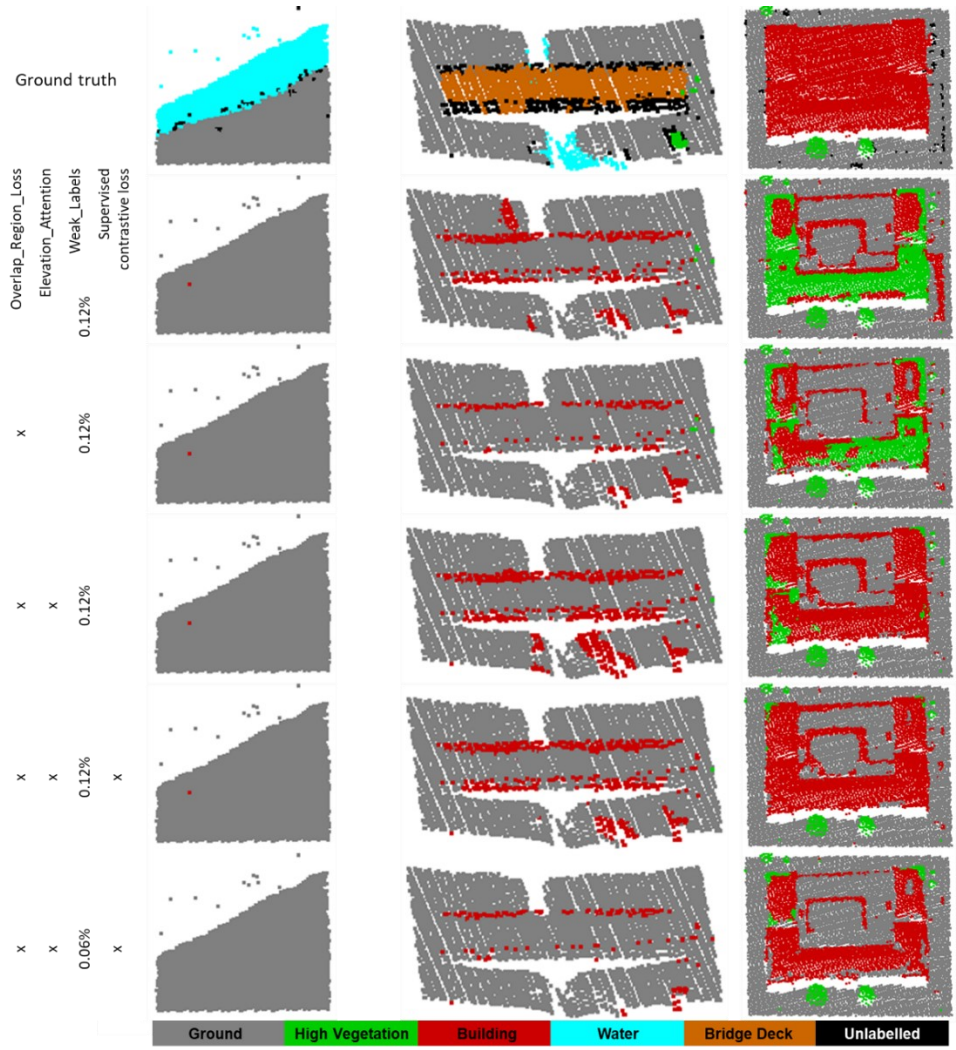


Figure 4.25 Qualitative results on the DFC2019 testing data.

Classification results on the DFC2019 testing data obtained by different training pseudo labels are demonstrated in Table 4.18. The overlap region loss in the classification network gives rise to the increases of the average F1 score and the overall accuracy by 0.023 and 0.02 respectively. More accurate building points in pseudo labels brought by the elevation attention directly contribute to better predictions on building points for the testing data. When training the segmentation network with the supervised contrastive loss, our results are further improved on high vegetation and building points. Under the condition of 0.06% weak labels, our results are better than the baseline using 0.12% weak labels.

4.5 Conclusion

This chapter investigates how to use weak labels on subclouds in order to supervise the training for semantic segmentation of ALS point clouds. Our method has two steps, pseudo label generation and training of semantic segmentation networks. For the first step, we use the PCAM from the classification network to induce localization cues for different categories. We propose an overlap region loss to make use of the semantic information inferred from overlapping subclouds. Then an elevation attention unit is designed to allow the classification network to produce more precise pseudo labels. For the training of the segmentation network, a supervised contrastive loss is adopted to help the network to learn from inaccurate pseudo labels. Our experiments on three ALS datasets show the effectiveness of our proposed method.

Since we only weakly label subclouds, precise localization information is not available for the training. This leads to the failure of small objects like powerlines and small bridge decks, compared to the fully supervised networks. For future work, an active learning strategy could be involved to only annotate those informative subclouds to further reduce the labelling efforts. Since weak subcloud labels have a limited ability to provide localization cues for small objects, some pointwise labels could be introduced under a limited annotation budget. Our work adopts a two-step pipeline, while an end-to-end trained network that can be directly supervised by weakly labelled subclouds and at the same time produce pointwise semantic labels should be further investigated.

Chapter 5 – Synthesis[4]

[4] This chapter is based on the previous chapters including lessons learned in working on the following publications:

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2021. Local and global encoder network for semantic segmentation of Airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 176, 151–168.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2020. Active and incremental learning for semantic ALS point cloud segmentation. ISPRS Journal of Photogrammetry and Remote Sensing 169, 73–92.

Lin, Y., Vosselman, G., Yang, M.Y., 2022. Weakly supervised semantic segmentation of airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 187, 79–100.

5.1 Conclusions per objective

Semantic segmentation of ALS point clouds

This objective is addressed in Chapter 2. A network based on KPConv is proposed to learn characteristics of ALS point clouds. We allow the network to learn representations from local to global in order to reveal spatial contextual information from different scales.

The proposed LGEnet takes both 2D and 3D convolutions as local encoders. The 2D convolutions are capable of capturing features for points distributed in horizontal dimensions. Furthermore, contextual information is explored at the object level through the proposed segment-based Edge Conditioned Convolution (SegECC), where graphs are constructed among segments. Then a spatial-channel attention is placed at the end of the network. The spatial attention models the global interdependencies between points by calculating the pairwise correlations between all points within an input sphere. The channel attention estimates the similarities between channels, aiming to enhance the learning of class specific discriminative features.

The advantages of the proposed LGENet are verified on two ALS datasets, the ISPRS benchmark dataset and the DFC2019 dataset. LGENet outperforms the baseline method (KPConv) on both datasets. Experiments on the ISPRS benchmark dataset show that the combination of 2D and 3D convolutions enables the network to learn more discriminative features for elongated objects distributed on horizontal planes. Also, networks with SegECC layers give better predictions on fine structures in both datasets, powerlines and fence/hedge in the ISPRS benchmark dataset and the bridge deck in the DFC2019 dataset. The spatial-channel attention further boosts the network performance and makes the LGENet achieve the state-of-the-art results attained in 2020.

Active and incremental learning for semantic segmentation of ALS point clouds

This objective is addressed in Chapter 3. Intending to reduce annotation efforts, we propose an active and incremental learning framework in which informative samples are iteratively selected and annotated to train deep learning models for the semantic segmentation of ALS point clouds. The deep learning models incrementally learn from the updated labelled pool.

In our proposed framework, we iteratively select point cloud tiles from the unlabelled pool and then incrementally enrich the model

knowledge. For the evaluation criteria, we implement two data dependent uncertainty metrics (point entropy and segment entropy) and one model dependent metric (mutual information). Compared to the point entropy that only evaluates individual points, our proposed segment entropy estimates the semantic heterogeneity within geometrical homogenous units. We assess the informativeness of point cloud tiles at the segment level because points in a geometrical homogenous unit are supposed to share the same label. If different labels are predicted within a segment, that segment should be selected to enrich the model knowledge in the next training. We also use mutual information to assess the disagreements in model predictions caused by the uncertainty of model parameters. For the training stage, except for the initial model that is trained from scratch, we fine-tune the previous network on the enlarged labelled dataset in order to save the training time.

Our proposed framework is tested on two ALS datasets. Our experimental results suggest that all evaluation metrics can select informative point cloud tiles that benefit the model training, compared to the random selection. The segment entropy gives the best performance for the ALS datasets. It achieves the full training accuracy by only taking a subset of the entire training data. The proportion is 31.7% for the Rotterdam dataset and 9% for the Amsterdam dataset. Also, compared to the training from scratch for each iteration, our fine-tuning strategy saves about half of the training time without sacrificing the model performance.

Weak supervision on semantic segmentation of ALS point clouds

This objective is addressed in Chapter 4. The idea is to supervise the training for semantic segmentation of ALS point clouds by weak subcloud labels. There are two basic steps to achieve this goal. The first step is to train a classification network with weak subcloud labels after which the pointwise pseudo labels on the training data are produced by the trained classification network. The second step is to exploit the produced pseudo labels to train a segmentation network which then produces predictions on the testing data.

To boost the performance of the classification network, we first design an overlap region loss to explore the semantic heterogeneity within a subcloud. Since subclouds are extracted with overlaps to make sure the entire training area is fully exploited during the training, the overlap region loss considers the inferred semantic labels on the overlapped region. This introduces more localization cues to the classification

network and these cues allow the classification network to produce more accurate pointwise pseudo labels on the training data. Since elevation related features are proven to be a critical element in the semantic segmentation of ALS point clouds, we design an elevation attention that is placed at the end of the classification network in order to allow the classification network to learn more representative features from ALS data. More accurate pseudo labels can be generated by the improved classification network. For the training of the segmentation network, we use a supervised contrastive loss that uncovers the underlying correlations of class-specific features. This loss allows the segmentation network to effectively learn more representative class specific features from inaccurate pointwise pseudo labels.

Experiments on three ALS datasets show the superior performance of our proposed model to the baseline method (MPRM). Our results prove that the localization cues introduced by the inferred semantic labels from the overlapping subclouds are effective in producing better pseudo labels. Also, elevation related features are still of importance to the training for the semantic segmentation of ALS point clouds. The supervised contrastive loss shows its ability to enhance the learning from the pseudo labels. The benefit is maximal when the network dynamically updates the labels for those points whose pseudo labels generated by the classification network are not confident.

The limitation of the weak supervision by subcloud labels is the lack of precise localization cues for different semantic categories. This causes the failures on the small objects in the scenes. Therefore, with a limited annotation budget, pointwise labels could be included for those critical points to further improve the model performance.

5.2 Reflections and outlook

Dynamic graph construction

In this thesis, we extracted contextual information from local to global. For the contextual information at the object level, we first partitioned point clouds into segments through an unsupervised algorithm and then the segments were kept the same during the training. The main advantage of these fixed segments is that they are quite computationally efficient. Every time we constructed graphs for SegECC, we did not have to cluster points again. Although DGCNN (Wang et al., 2019b) can dynamically update graphs during the training, it clusters points in the feature space in each forward pass which is highly computationally expensive. The high computational cost

makes this kind of dynamic graph infeasible to process large scale point clouds over urban areas.

In spite of the computational efficiency, the fixed segmentation results still have some limitations. If objects with different semantic labels are clustered into the same segment, the network cannot correct the mistake and this mistake will mislead the network during the training. Also, our results in Table 2.5 suggested that the performance of SegECC was influenced by the segmentation results. Inappropriate segmentation results even degraded the network performance. These limitations suggest that a computationally efficient approach to dynamically clustering points is required to further improve the network performance.

Spatial Transformer (Wang et al., 2021) which aims at learning geometric transformations to the point clouds provides a possible solution. Instead of clustering points in the high dimensional feature space, the Spatial Transformer first learns the geometric transformation matrices and then clusters the transformed points in the 3D space. The learned transformation matrices are conditioned on point coordinates and features and they can pull together points that share similar geometrical features in the 3D space. Therefore, retrieving neighbours in the transformed point clouds will include points that have similar features but are far away in the original point clouds. Spatial Transformer provides a new perspective to efficiently cluster points and this could be further investigated in the neural networks to facilitate the learning of representative features.

Self-attention for global features

To encode the global contextual information, we applied the spatial-channel attention which was first proposed by Fu et al. (2018) in the image semantic segmentation. The spatial-channel attention was placed at the end of the network to perceive all important class-specific information. The major limitation of the spatial attention is the high computational complexity since it has to calculate the interdependencies between all possible point pairs within the input point cloud.

One possible solution is to apply the spatial attention to the downsampled point clouds with multiple scales and then attentively aggregate these output features (Xu et al., 2022). Since the number of points is reduced in low-resolution data, the computation of the self-attention on the downsampled points is much more efficient compared to the calculation on the original data. Except for grid sampling (Xu et

al., 2022), LighTN (Wang et al., 2022) saves computational resources of the self-attention based operation by using a differentiable sampling operation to find the most representative subsets. The sampling operation can be trained end-to-end to learn from the data. Global contextual information is encoded based on representative subsets. LighTN also reduces the computational cost without sacrificing the accuracy by applying a simplified self-attention operation, called self-correlation, where the three projection layers in the original self-attention operation are removed. In the future, the efficiency of using the self-attention operations to embed global contextual information can be improved through the appropriate downsampling operations and more lightweight structures.

Self-attention as local feature encoders

Apart from using self-attention based operations to encode global features in Chapter 2, attention modules were also introduced to weakly supervised training based on subcloud labels in Chapter 4. Not only did the baseline method MPRM design three attention heads to mine localization cues, but we also proposed an elevation attention aiming to enhance the learning of class-specific features. These self-attention based modules were also designed to encode global features. The self-attention operators can also be applied to encode local features which are taken as the basic feature extractor replacing convolutions. Since point clouds are unevenly distributed, the permutation invariance of the self-attention makes it a solution to effectively extract point cloud features. Some recent works show the potential of the self-attention based encoder. PCT (Guo et al., 2021) builds a base network upon the self-attention and the base network achieves good performance on a set of downstream tasks, including classification, segmentation and normal estimation. Zhao et al. (2021) build residual blocks upon point transformer layers and construct the segmentation network following the U-net design where points are downsampled in the encoder and upsampled in the decoder. In the future, self-attention could be further investigated to learn the representative local features for ALS point clouds.

Training with fewer annotation efforts

Pointwise annotation of point clouds for the training of semantic segmentation networks requires human labour. This thesis alleviated the required annotation efforts through active learning (Chapter 3) and weakly supervised learning (Chapter 4). In the active learning strategy, we selected tiles and annotated all points within the selected tiles. However, some points may be less informative and have a limited

capacity for enriching the network knowledge. The selection could be further optimized by ignoring the less informative data and only focusing on the most informative clusters (Shi et al., 2021; Wu et al., 2021; Xie et al., 2020). Although subcloud labels are much cheaper compared to the pointwise annotation and our proposed overlap region loss can provide more exact localization cues, our results in Chapter 4 have shown that they are still difficult to use to supervise the network to learn delicate structures such as powerlines. In the future, pointwise annotation could be actively added to the weak supervision on subcloud labels with a limited annotation budget.

Both active learning and weakly supervised learning aim to make the network learn representative features from the data with limited annotation. Unsupervised pre-training aiming to learn internal data structures through various designed tasks that do not require semantic labels is one of the potential solutions to save the annotation cost. In the pre-training, the network can be trained to recover broken point clouds (Sauder and Sievers, 2019), set pointwise correspondences between point clouds captured from different views of the same scene (Xie et al., 2020) or complete the point clouds of partially occluded objects (Wang et al., 2020). Through these designed tasks, networks explore the structures of point clouds and intermediate representatives can be learned from the data. With the learned latent information, pre-trained networks only need a small number of semantic labels for the semantic segmentation without sacrificing the network performance.

Although deep learning networks are capable of learning representative features from the data, they may fail to generalize to various inputs with changes in data distribution. This may happen when ALS point clouds are captured from different cities or with different point densities. In this thesis, the pointwise predictions on two Dutch cities, Rotterdam and Amsterdam, were produced by the models separately trained on their subsets and labels were provided to both cities. To save annotation efforts, how to transfer the existing knowledge from one dataset to another without extra annotations need to be investigated. Domain adaption aiming to bridge the gap between two datasets in the feature space is a possible strategy (Achituve et al., 2021). With this strategy, the model trained on the Rotterdam dataset can be applied to the Amsterdam dataset without extra labelling efforts.

Bibliography

Achituve, I., Maron, H., Chechik, G., 2021. Self-Supervised Learning for Domain Adaptation on Point-Clouds, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 123–133.

Arief, H.A.A., Indahl, U.G., Strand, G.H., Tveite, H., 2019. Addressing overfitting on point cloud classification using Atrous XCRF. ISPRS Journal of Photogrammetry and Remote Sensing 155, 90–101.

Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S., 2016. 3D semantic parsing of large-scale indoor spaces, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 1534–1543.

Bai, X., Ren, P., Zhang, H., Zhou, J., 2015. An incremental structured part model for object recognition. Neurocomputing 154, 189–199.

Beluch, W.H., Genewein, T., Nürnberger, A., Köhler, J.M., 2018. The Power of Ensembles for Active Learning in Image Classification, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 9368–9377.

Bosch, M., Foster, K., Christie, G., Wang, S., Hager, G.D., Brown, M., 2019. Semantic Stereo for Incidental Satellite Images, in: IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, pp. 1524–1532.

Boulch, A., 2020. ConvPoint: Continuous Convolutions for Point Cloud Processing. Computers and Graphics 88, 24–34.

Boulch, A., Guerry, J., Le Saux, B., Audebert, N., 2018. SnapNet: 3D Point Cloud Semantic Labeling with 2D Deep Segmentation Networks. Computers and Graphics (Pergamon) 71, 189–198.

Boulch, A., Puy, G., Marlet, R., 2020. FKAConv: Feature-Kernel Alignment for Point Cloud Convolution, in: Proceedings of the Asian Conference on Computer Vision.

Breiman, L., 2001. Random Forests. Machine Learning 45, 5–32.

Breiman, L., 1996. BIAS, VARIANCE , AND ARCING CLASSIFIERS.

Brust, C.-A., Käding, C., Denzler, J., 2020. Active and Incremental Learning with Weak Supervision. KI - Künstliche Intelligenz 1–16.

Castro, F.M., Marín-Jiménez, M.J., Guil, N., Schmid, C., Alahari, K., 2018. End-to-End Incremental Learning, in: Proceedings of the European Conference on Computer Vision (ECCV). pp. 233–248.

Chang, Y.-T., Wang, Q., Hung, W.-C., Piramuthu, R., Tsai, Y.-H., Yang, M.-H., 2020. Mixup-CAM: Weakly-supervised Semantic Segmentation via Uncertainty Regularization, in: British Machine Vision Conference (BMVC).

Chehata, N., Guo, L., Mallet, C., 2009. Airborne Lidar Feature Selection for Urban Classification Using Random Forests, in: International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences.

Chen, L., Chen, W., Xu, Z., Huang, H., Wang, S., Zhu, Q., Li, H., 2021. DAPnet: A Double Self-Attention Convolutional Network for Point Cloud Semantic Labeling. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 14, 9680–9691.

Chen, Z., Gao, B., Devereux, B., 2017. State-of-the-Art: DTM Generation Using Airborne LIDAR Data. Sensors 17.

Cooper, H.M., Chen, Q., Fletcher, C.H., Barbee, M.M., 2013. Assessing

Vulnerability Due to Sea-Level Rise in Maui, Hawai'i Using Lidar Remote Sensing and GIS. Climatic Change 116, 547–563.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Machine Learning 20, 273–297.

Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M., 2017. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017. Institute of Electrical and Electronics Engineers Inc., pp. 2432–2443.

Dai, J., He, K., Sun, J., 2015. BoxSup: Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation, in: Proceedings of the IEEE International Conference on Computer Vision. pp. 1635–1643.

Deng, A., Wu, Y., Zhang, P., Lu, Z., Li, W., Su, Z., 2022. A weakly supervised framework for real-world point cloud classification. Computers & Graphics 102, 78–88.

Dou, J., Li, J., Qin, Q., Tu, Z., 2015. Robust visual tracking based on incremental discriminative projective non-negative matrix factorization. Neurocomputing 166, 210–228.

Fan, J., Zhang, Z., Tan, T., Song, C., Xiao, J., 2020. CIAN: Cross-Image Affinity Net for Weakly Supervised Semantic Segmentation, in: Proceedings of the AAAI Conference on Artificial Intelligence. AAAI press, pp. 10762–10769.

Feng, D., Wei, X., Rosenbaum, L., Maki, A., Dietmayer, K., 2019. Deep Active Learning for Efficient Training of a LiDAR 3D Object Detector. arXiv preprint arXiv:1901.10609.

Feng, M., Zhang, L., Lin, X., Gilani, S.Z., Mian, A., 2020. Point attention network for semantic segmentation of 3D point clouds. Pattern Recognition 107, 107446.

Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., Lu, H., 2018. Dual Attention Network for Scene Segmentation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2019-June, 3141–3149.

Gal, Y., Ghahramani, Z., 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning Zoubin Ghahramani, in: International Conference on Machine Learning. pp. 1050–1059.

Gal, Y., Islam, R., Ghahramani, Z., 2017. Deep Bayesian Active Learning with Image Data, in: In Proceedings of the 34th International Conference on Machine Learning. pp. 1183–1192.

Gerke, M., Xiao, J., 2013. Supervised and unsupervised MRF based 3d scene classification in multiple view airborne oblique images.

Graham, B., Engelcke, M., Van Der Maaten, L., 2018. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 9224–9232.

Groh, F., Wieschollek, P., Lensch, H.P.A., 2019. Flex-Convolution: Million-Scale Point-Cloud Learning Beyond Grid-Worlds, in: Asian Conference on Computer Vision. Springer, pp. 105–122.

Guo, M.-H., Cai, J.-X., Liu, Z.-N., Mu, T.-J., Martin, R.R., Hu, S.-M., 2021. PCT: Point cloud transformer. Computational Visual Media 7, 187–199.

Hou, J., Graham, B., Nießner, M., Xie, S., 2021. Exploring Data-Efficient 3D Scene Understanding with Contrastive Scene Contexts, in: Proceedings

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15587–15597.

Hou, Q., Jiang, P.T., Wei, Y., Cheng, M.M., 2018. Self-Erasing Network for Integral Object Attention, in: Advances in Neural Information Processing Systems. pp. 549–559.

Hu, J., Shen, L., Sun, G., 2018. Squeeze-and-Excitation Networks, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 7132–7141.

Hu, Q., Ang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A., 2020. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11108–11117.

Hu, Q., Yang, B., Fang, G., Guo, Y., Leonardis, A., Trigoni, N., Markham, A., 2021. SQN: Weakly-Supervised Semantic Segmentation of Large-Scale 3D Point Clouds with 1000x Fewer Labels. arXiv preprint arXiv:2104.04891.

Hu, X., Yuan, Y., 2016. Deep-Learning-Based Classification for DTM Extraction from ALS Point Cloud. Remote Sensing 8, 730.

Huang, R., Xu, Y., Hong, D., Yao, W., Ghamisi, P., Stilla, U., 2020. Deep point embedding for urban classification using ALS point clouds: A new perspective from local to global. ISPRS Journal of Photogrammetry and Remote Sensing 163, 62–81.

Huang, R., Xu, Y., Stilla, U., 2021. GraNet: Global relation-aware attentional network for semantic segmentation of ALS point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 177, 1–20.

Huang, W., Sun, M., Li, S., 2016. A 3D GIS-based interactive registration mechanism for outdoor augmented reality system. Expert Systems with Applications 55, 48–58.

Jiang, M., Wu, Y., Zhao, T., Zhao, Z., Lu, C., 2018. PointSIFT: A SIFT-like Network Module for 3D Point Cloud Semantic Segmentation. arXiv preprint arXiv:1807.00652.

Kalogerakis, E., Averkiou, M., Maji, S., Chaudhuri, S., 2017. 3D Shape Segmentation with Projective Convolutional Networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3779–3788.

Kellenberger, B., Marcos, D., Lobry, S., Tuia, D., 2019. Half a Percent of Labels is Enough: Efficient Animal Detection in UAV Imagery using Deep CNNs and Active Learning. IEEE Transactions on Geoscience and Remote Sensing 57, 9524–9533.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D., 2020. Supervised Contrastive Learning, in: Advances in Neural Information Processing Systems. Neural information processing systems foundation, pp. 18661–18673.

Kingma, D.P., Ba, J.L., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., Hadsell, R., 2017. Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences 114, 3521–3526.

Kohli, P., Ladický, L., Torr, P.H.S., 2009. Robust Higher Order Potentials for

Enforcing Label Consistency. International Journal of Computer Vision 82, 302–324.

Kölle, M., Walter, V., Schmohl, S., Soergel, U., 2021. Remembering Both the Machine and the Crowd When Sampling Points: Active Learning for Semantic Segmentation of ALS Point Clouds, in: International Conference on Pattern Recognition. Springer, Cham, pp. 505–520.

Landrieu, L., Obozinski, G., 2017. Cut Pursuit: Fast Algorithms to Learn Piecewise Constant Functions on General Weighted Graphs. SIAM Journal on Imaging Sciences 10, 1724–1766.

Landrieu, L., Raguet, H., Vallet, B., Mallet, C., Weinmann, M., 2017. A structured regularization framework for spatially smoothing semantic labelings of 3D point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 132, 102–118.

Landrieu, L., Simonovsky, M., 2018. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4558–4567.

Lemmen, C., van Oosterom, P., Bennett, R., 2015. The Land Administration Domain Model. Land Use Policy 49, 535–545.

Li, K., Wu, Z., Peng, K.C., Ernst, J., Fu, Y., 2020. Guided Attention Inference Network. IEEE Transactions on Pattern Analysis and Machine Intelligence 42, 2996–3010.

Li, W., Wang, F., Xia, G., 2020. A geometry-attentional network for ALS point cloud classification. ISPRS Journal of Photogrammetry and Remote Sensing 164, 26–40.

Li, X., Wang, L., Wang, M., Wen, C., Fang, Y., 2020. DANCE-NET: Density-aware convolution networks with context encoding for airborne LiDAR point cloud classification. ISPRS Journal of Photogrammetry and Remote Sensing 166, 128–139.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B., 2018. PointCNN: Convolution On X-Transformed Points, in: Advances in Neural Information Processing Systems. pp. 820–830.

Li, Y., Hu, Q., Wu, M., Liu, J., Wu, X., 2016. Extraction and Simplification of Building Façade Pieces from Mobile Laser Scanner Point Clouds for 3D Street View Services. ISPRS International Journal of Geo-Information 5, 231.

Lin, C.-H., Chen, J.-Y., Su, P.-L., Chen, C.-H., 2014. Eigen-Feature Analysis of Weighted Covariance Matrices for LiDAR Point Cloud Classification. ISPRS Journal of Photogrammetry and Remote Sensing 94, 70–79.

Lin, D., Dai, J., Jia, J., He, K., Sun, J., 2016. ScribbleSup: Scribble-Supervised Convolutional Networks for Semantic Segmentation, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 3159–3167.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2021. Local and global encoder network for semantic segmentation of Airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 176, 151–168.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2020a. Active and incremental learning for semantic ALS point cloud segmentation. ISPRS Journal of Photogrammetry and Remote Sensing 169, 73–92.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2020b. Efficient Training of Semantic Point Cloud Segmentation via Active Learning, in: ISPRS Annals

of the Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 243–250.

Lin, Y., Yang, M.Y., Nex, F., 2018. Semantic Building Façade Segmentation from Airborne Oblique Images, in: ISPRS TC II Mid-Term Symposium.

Lodha, S.K., Fitzpatrick, D.M., Helmbold, D.P., 2007. Aerial Lidar Data Classification using AdaBoost, in: Sixth International Conference on 3-D Digital Imaging and Modeling (3DIM 2007). IEEE, pp. 435–442.

Lodha, S.K., Kreps, E.J., Helmbold, D.P., Fitzpatrick, D., 2006. Aerial LiDAR Data Classification Using Support Vector Machines (SVM), in: Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06). IEEE, pp. 567–574.

Luo, H., Wang, C., Wen, C., Chen, Z., Zai, D., Yu, Y., Li, J., 2018. Semantic Labeling of Mobile LiDAR Point Clouds via Active Learning and Higher Order MRF. IEEE Transactions on Geoscience and Remote Sensing 56, 3631–3644.

Mao, J., Wang, X., Li, H., 2019. Interpolated Convolutional Networks for 3D Point Cloud Understanding, in: Proceedings of the IEEE International Conference on Computer Vision. pp. 1578–1587.

Maturana, D., Scherer, S., 2015. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 922–928.

McCallumzy, A.K., Nigamy, K., 1998. Employing EM and Pool-Based Active Learning for Text Classiication, in: Proc. International Conference on Machine Learning (ICML). Citeseer, pp. 359–367.

Meng, X., Currit, N., Wang, L., Yang, X., 2012. Detect Residential Buildings from Lidar and Aerial Photographs through Object-Oriented Land-Use Classification. Photogrammetric Engineering & Remote Sensing 78, 35–44.

Murgante, B., Borruso, G., Lapucci, A., 2009. Geocomputation and Urban Planning, in: Geocomputation and Urban Planning. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1–17.

Murtha, T., Golden, C., Cyphers, A., Klippel, A., Flohr, T., 2018. Beyond Inventory and Mapping: LIDAR, Landscape and Digital Landscape Architecture. Journal of Digital Landscape Architecture 3, 249–259.

Najafi, M., Taghavi Namin, S., Salzmann, M., Petersson, L., 2014. Non-associative Higher-Order Markov Networks for Point Cloud Classification, in: European Conference on Computer Vision. Springer, Cham, pp. 500–515.

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

Niemeyer, J., Rottensteiner, F., Soergel, U., Heipke, C., 2016. Hierarchical Higher Order CRF for the Classification of Airborne Lidar Point Clouds in Urban Areas, in: ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 655–662.

Niemeyer, J., Wegner, J.D., Mallet, C., Rottensteiner, F., Soergel, U., 2011. Conditional random fields for urban scene classification with full waveform LiDAR data, in: ISPRS Conference on Photogrammetric Image Analysis. Springer, Berlin, pp. 233–244.

Oh, S.J., Benenson, R., Khoreva, A., Akata, Z., Fritz, M., Schiele, B., 2017. Exploiting saliency for object segmentation from image level labels, in:

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5038–5047.

Otálora, S., Perdomo, O., González, F., Müller, H., 2017. Training Deep Convolutional Neural Networks with Active Learning for Exudate Classification in Eye Fundus Images, in: Intravascular Imaging and Computer Assisted Stenting, and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis. Springer, pp. 146–154.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Xamla, A.K., Yang, E., Devito, Z., Raison Nabla, M., Tejani, A., Chilamkurthy, S., Ai, Q., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library, in: Advances in Neural Information Processing Systems. pp. 8026–8037.

Pham, Q.-H., Thanh Nguyen, D., Hua, B.-S., Roig, G., Yeung, S.-K., 2019. JSIS3D: Joint Semantic-Instance Segmentation of 3D Point Clouds with Multi-Task Pointwise Networks and Multi-Value Conditional Random Fields, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8827–8836.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 652–660.

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space, in: Advances in Neural Information Processing Systems. pp. 5099–5108.

Ristin, M., Guillaumin, M., Gall, J., Van Gool, L., 2016. Incremental Learning of Random Forests for Large-Scale Image Classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 38, 490–503.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision 115, 211–252.

Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R., 2016. Progressive Neural Networks. arXiv preprint arXiv:1606.04671.

Sauder, J., Sievers, B., 2019. Self-Supervised Deep Learning on Point Clouds by Reconstructing Space. Advances in Neural Information Processing Systems 32.

Schmohl, S., Sörgel, U., 2019. Submanifold Sparse Convolutional Networks for Semantic Segmentation of Large-Scale ALS Point Clouds, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 77–84.

Settles, B., 2009. Active Learning Literature Survey. University of Wisconsin-Madison Department of Computer Sciences.

Settles, B., Craven, M., 2008. An Analysis of Active Learning Strategies for Sequence Labeling Tasks, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics.

Shen, Y., Wu, L., Wang, Z., 2010. Identification of Inclined Buildings from Aerial LiDAR Data for Disaster Management, in: 2010 18th International Conference on Geoinformatics. IEEE, pp. 1–5.

Shi, X., Xu, X., Chen, K., Cai, L., Foo, C.S., Jia, K., 2021. Label-Efficient Point Cloud Semantic Segmentation: An Active Learning Approach. arXiv preprint arXiv:2101.06931.

Shin, G., Yooun, H., Shin, D., Shin, D.K., 2018. Incremental learning method for cyber intelligence, surveillance, and reconnaissance in closed military network using converged IT techniques. Soft Computing 22, 6835–6844.

Simonovsky, M., Komodakis, N., 2017. Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs. Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 2017-January, 29–38.

Song, C., Huang, Y., Ouyang, W., Wang, L., 2019. Box-Driven Class-Wise Region Masking and Filling Rate Guided Loss for Weakly Supervised Semantic Segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3136–3145.

Stammes, E., Runia, T.F.H., Hofmann, M., Ghafoorian, M., 2021. Find it if You Can: End-to-End Adversarial Erasing for Weakly-Supervised Semantic Segmentation, in: Thirteenth International Conference on Digital Image Processing (ICDIP 2021). International Society for Optics and Photonics, p. 12.

Tao, A., Duan, Y., Wei, Y., Lu, J., Zhou, J., 2020. SegGroup: Seg-Level Supervision for 3D Instance and Semantic Segmentation. arXiv preprint arXiv:2012.10217.

Tasar, O., Tarabalka, Y., Alliez, P., 2018. Incremental Learning for Semantic Segmentation of Large-Scale Remote Sensing Data. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 12, 3524–3537.

Tchapmi, L., Choy, C.B., Armeni, I., Gwak, J., Savarese, S., 2017. SEGCloud: Semantic Segmentation of 3D Point Clouds, in: 2017 International Conference on 3D Vision (3DV). IEEE, pp. 537–547.

Thomas, H., Qi, C.R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L.J., 2019. KPConv: Flexible and Deformable Convolution for Point Clouds, in: Proceedings of the IEEE International Conference on Computer Vision. pp. 6411–6420.

Tombari, F., Salti, S., Di Stefano, L., 2010. Unique signatures of histograms for local surface description, in: European Conference on Computer Vision. Springer Verlag, pp. 356–369.

Tuia, D., Volpi, M., Copa, L., Kanevski, M., Munoz-Mari, J., 2011. A Survey of Active Learning Algorithms for Supervised Remote Sensing Image Classification. IEEE Journal of Selected Topics in Signal Processing 5, 606–617.

Varney, N., Asari, V.K., Graehling, Q., 2020. Pyramid Point: A Multi-Level Focusing Network for Revisiting Feature Layers. arXiv preprint arXiv:2011.08692.

Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention Is All You Need, in: Advances in Neural Information Processing Systems. pp. 5998–6008.

Vezhnevets, A., Buhmann, J.M., Ferrari, V., 2012. Active Learning for Semantic Segmentation with Expected Change, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, pp. 3162–3169.

Vosselman, G., Coenen, M., Rottensteiner, F., 2017. Contextual segment-based classification of airborne laser scanner data. ISPRS Journal of

Photogrammetry and Remote Sensing 128, 354–371.
Vosselman, G., Maas, H.-G., 2010. Airborne and terrestrial laser scanning. CRC Press.
Wallace, L., Lucieer, A., Watson, C., Turner, D., Wallace, L., Lucieer, A., Watson, C., Turner, D., 2012. Development of a UAV-LiDAR System with Application to Forest Inventory. Remote Sensing 4, 1519–1543.
Wang, H., Liu, Q., Yue, X., Lasenby, J., Kusner, M.J., 2020. Unsupervised Point Cloud Pre-Training via Occlusion Completion, in: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9782–9792.
Wang, H., Rong, X., Yang, L., Wang, S., Tian, Y., 2019. Weakly Supervised Semantic Segmentation in 3D Graph-Structured Point Clouds of Wild Scenes, in: 30th British Machine Vision Conference 2019 (BMVC 2019). BMVA Press.
Wang, J., Chakraborty, R., Yu, S.X., 2021. Spatial Transformer for 3D Point Clouds. IEEE Transactions on Pattern Analysis and Machine Intelligence.
Wang, K., Zhang, D., Li, Y., Zhang, R., Lin, L., 2017. Cost-Effective Active Learning for Deep Image Classification. IEEE Transactions on Circuits and Systems for Video Technology 27, 2591–2600.
Wang, P., Yao, W., 2021. A new weakly supervised approach for ALS point cloud semantic segmentation. arXiv preprint arXiv:2110.01462.
Wang, R., Albooyeh, M., Ravanbakhsh, S., 2020. Equivariant Maps for Hierarchical Structures. arXiv preprint arXiv:2006.03627.
Wang, X., Girshick, R., Gupta, A., He, K., 2018. Non-local Neural Networks, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 7794–7803.
Wang, X., Jin, Y., Cen, Y., Wang, T., Tang, B., Li, Y., 2022. LighTN: Light-weight Transformer Network for Performance-overhead Tradeoff in Point Cloud Downsampling. arXiv preprint arXiv:2202.06263.
Wang, Y., Mendez Mendez, A.E., Cartwright, M., Bello, J.P., 2019a. Active Learning for Efficient Audio Annotation and Classification with a Large Amount of Unlabeled Data, in: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 880–884.
Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M., 2019b. Dynamic Graph CNN for Learning on Point Clouds. ACM Transactions on Graphics (TOG) 38, 1–12.
Wei, J., Lin, G., Yap, K.H., Hung, T.Y., Xie, L., 2020. Multi-Path Region Mining For Weakly Supervised 3D Semantic Segmentation on Point Clouds, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 4383–4392.
Wei, Y., Feng, J., Liang, X., Cheng, M.M., Zhao, Y., Yan, S., 2017. Object Region Mining with Adversarial Erasing: A Simple Classification to Semantic Segmentation Approach, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Institute of Electrical and Electronics Engineers Inc., pp. 1568–1576.
Weinmann, M., Jutzi, B., Hinz, S., Mallet, C., 2015. Semantic Point Cloud Interpretation Based on Optimal Neighborhoods, Relevant Features and Efficient Classifiers. ISPRS Journal of Photogrammetry and Remote Sensing 105, 286–304.
Weinmann, M., Jutzi, B., Mallet, C., 2014. Semantic 3D Scene Interpretation: A Framework Combining Optimal Neighborhood Size Selection with

Relevant Features, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 181–188.

Weinmann, M., Jutzi, B., Mallet, C., 2013. Feature Relevance Assessment for the Semantic Interpretation of 3D Point Cloud Data, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences.

Wen, C., Li, X., Yao, X., Peng, L., Chi, T., 2021. Airborne LiDAR point cloud classification with global-local graph attention convolution neural network. ISPRS Journal of Photogrammetry and Remote Sensing 173, 181–194.

Wen, C., Yang, L., Li, X., Peng, L., Chi, T., 2020. Directionally constrained fully convolutional neural network for airborne LiDAR point cloud classification. ISPRS Journal of Photogrammetry and Remote Sensing 162, 50–62.

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

Wu, T.-H., Liu, Y.-C., Huang, Y.-K., Lee, H.-Y., Su, H.-T., Huang, P.-C., Hsu, W.H., 2021. ReDAL: Region-based and Diversity-aware Active Learning for Point Cloud Semantic Segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15510–15519.

Wu, W., Qi, Z., Fuxin, L., 2019. PointConv: Deep Convolutional Networks on 3D Point Clouds, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9621–9630.

Wu, Z., Song, S., Khosla, A., Fisher, Y., Zhang, L., Tang, X., Xiao, J., 2015. 3D ShapeNets: A Deep Representation for Volumetric Shapes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1912–1920.

Xie, S., Gu, J., Guo, D., Qi, C.R., Guibas, L., Litany, O., 2020. PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding, in: European Conference on Computer Vision. Springer, pp. 574–591.

Xiong, X., Munoz, D., Bagnell, J.A., Hebert, M., 2011. 3-D scene analysis via sequenced predictions over points and regions, in: Proceedings - IEEE International Conference on Robotics and Automation. pp. 2609–2616.

Xu, S., Vosselman, G., Oude Elberink, S., 2014. Multiple-Entity Based Classification of Airborne Laser Scanning Data in Urban Areas. ISPRS Journal of Photogrammetry and Remote Sensing 88, 1–15.

Xu, S., Wan, R., Ye, M., Zou, X., Cao, T., 2022. Sparse Cross-scale Attention Network for Efficient LiDAR Panoptic Segmentation. arXiv preprint arXiv:2201.05972.

Xu, X., Lee, G.H., 2020. Weakly Supervised Semantic Point Cloud Segmentation:Towards 10X Fewer Labels, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 13703–13712.

Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y., 2018. SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters, in: Proceedings of the European Conference on Computer Vision (ECCV). pp. 87–102.

Xu, Z., Akella, R., Zhang, Y., 2007. Incorporating diversity and density in active learning for relevance feedback, in: European Conference on Information Retrieval. Springer, Berlin, Heidelberg, pp. 246–257.

Yang, Z., Jiang, W., Lin, Y., Elberink, S.O., 2020. Using training samples retrieved from a topographic map and unsupervised segmentation for the

classification of airborne laser scanning data. Remote Sensing 12, 1–18.
Yang, Z., Jiang, W., Xu, B., Zhu, Q., Jiang, S., Huang, W., Yang, Z., Jiang, W., Xu, B., Zhu, Q., Jiang, S., Huang, W., 2017. A Convolutional Neural Network-Based 3D Semantic Labeling Method for ALS Point Clouds. Remote Sensing 9, 936.
Yang, Z., Tan, B., Pei, H., Jiang, W., 2018. Segmentation and Multi-Scale Convolutional Neural Network-Based Classification of Airborne Laser Scanner Data. Sensors 18, 3347.
Ye, S., Chen, D., Han, S., Liao, J., 2021. Learning with Noisy Labels for Robust Point Cloud Segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6443–6452.
Yousefhussien, M., Kelbe, D.J., Ientilucci, E.J., Salvaggio, C., 2018. A multi-scale fully convolutional network for semantic labeling of 3D point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 143, 191–204.
Yu, Z., Zhuge, Y., Lu, H., Zhang, L., 2019. Joint Learning of Saliency Detection and Weakly Supervised Semantic Segmentation, in: Proceedings of the IEEE International Conference on Computer Vision. Institute of Electrical and Electronics Engineers Inc., pp. 7222–7232.
Zhang, Y., Li, Z., Xie, Y., Qu, Y., Li, C., Mei, T., 2021. Weakly Supervised Semantic Segmentation for Large-Scale Point Cloud, in: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 3421–3429.
Zhao, H., Jiang, L., Jia, J., Torr, P., Koltun, V., 2021. Point Transformer, in: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16259–16268.
Zhao, R., Pang, M., Wang, J., 2018. Classifying airborne LiDAR point clouds via deep features learned by a multi-scale convolutional neural network. International Journal of Geographical Information Science 32, 960–979.
Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P.H.S., 2015. Conditional Random Fields as Recurrent Neural Networks, in: Proceedings of the IEEE International Conference on Computer Vision. pp. 1529–1537.
Zhou, K., Lindenbergh, R., Gorte, B., Zlatanova, S., 2020. LiDAR-guided dense matching for detecting changes and updating of buildings in Airborne LiDAR data. ISPRS Journal of Photogrammetry and Remote Sensing 162, 200–213.
Zhou, Z., Shin, J., Zhang, L., Gurudu, S., Gotway, M., Liang, J., 2017. Fine-tuning convolutional neural networks for biomedical image analysis: Actively and incrementally, in: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017. pp. 4761–4772.
Zhu, X., 2005. Semi-supervised learning with graphs. Carnegie Mellon University.
Zhu, X., Goldberg, A.B., 2009. Introduction to semi-supervised learning. Synthesis Lectures on Artificial Intelligence and Machine Learning 3, 1–130.
Zolanvari, S.M.I., Ruano, S., Rana, A., Cummins, A., da Silva, R.E., Rahbar, M., Smolic, A., 2019. DublinCity: Annotated LiDAR Point Cloud and its Applications, in: Proceedings of the 30th British Machine Vision Conference.

Summary

ALS data are essential data sources used to generate digital terrain models (DTM), 3D city models, landscape models and high precision maps. Semantic segmentation aiming to assign every point with a semantic label of ALS point clouds is of importance when generating those 3D products that have multiple categories and ask for detailed object geometry. Motivated by the top performance of deep learning algorithms on scene understanding tasks, this Ph.D. thesis investigates the semantic segmentation of ALS point clouds based on deep learning algorithms. We first explore how to learn representative features from ALS point clouds (Chapter 2). Then we focus on how to reduce the manual labelling efforts to train a deep learning model for semantic segmentation. We investigate active learning (Chapter 3) to select and annotate informative points, and weak supervision (Chapter 4) to annotate only weak labels for the pointwise prediction task.

To allow the deep learning networks to learn representative features from ALS data and involve different levels of neighbouring information to extract pointwise geometrical features, we first designed a local feature extractor and then explored contextual information at both object and global levels. The proposed LGEnet takes both 2D and 3D convolutions as local encoders. The combination of 2D and 3D convolutions enables the network to learn more discriminative features for elongated objects distributed on horizontal planes. Furthermore, contextual information is explored at the object level through the proposed segment-based Edge Conditioned Convolution (SegECC), where graphs are constructed among segments. Then a spatial-channel attention is placed at the end of the network. The spatial attention models the global interdependencies between points by calculating the pairwise correlations between all points within an input sphere. The channel attention estimates the similarities between channels, aiming to enhance the learning of class specific discriminative features.

In order to reduce the required annotation efforts for the training of deep learning models, we propose an active and incremental learning framework for semantic segmentation of ALS point clouds. In this framework, we iteratively select point cloud tiles from the unlabelled pool and then incrementally enrich the model knowledge. For the selection criteria, we implement two data dependent uncertainty metrics (point entropy and segment entropy) and one model dependent metric (mutual information). Our proposed segment entropy estimates the semantic heterogeneity within geometrical homogenous units and it achieves the best performance for the ALS

datasets. Instead of training from scratch for each iteration, we fine-tune the previous network on the enlarged labelled dataset and this significantly saves the training time.

We also investigate how to alleviate the annotation efforts for the deep learning training through using weak subcloud labels instead of pointwise ground truth for ALS datasets. The first step is to train a classification network with weak subcloud labels after which the pointwise pseudo labels on the training data are produced by the trained classification network. The second step is to exploit the produced pseudo labels to train a segmentation network which then produces predictions on the testing data. The performance of the classification network is boosted by an overlap region loss and an elevation attention. The overlap region loss provides more localization cues to the classification network and the elevation attention allows the classification network to learn more representative features from ALS data. For the training of the segmentation network, we use a supervised contrastive loss that uncovers the underlying correlations of class-specific features. This loss allows the segmentation network to effectively learn more representative class specific features from inaccurate pointwise pseudo labels. The benefit is maximal when the network dynamically updates the labels for those points whose pseudo labels generated by the classification network are not confident.

In conclusion, this thesis investigates deep learning algorithms for semantic segmentation of ALS point clouds. One network is proposed to extract representative feature from ALS data and two methods are proposed to reduce the annotation efforts for the training of semantic segmentation networks. In future, how to use self-attention mechanisms as feature extractors could be further explored. Also, unsupervised pre-training and domain adaption are possible solutions to reduce required annotation efforts for the training of semantic segmentation networks.

Samenvatting

ALS-gegevens (Airborne Laser Scanning) zijn essentiële gegevensbronnen die gebruikt worden voor het genereren van digitale terreinmodellen (DTM), 3D-stadsmodellen, landschapsmodellen en zeer nauwkeurige kaarten. Semantische segmentatie, met als doel elk punt een semantisch label toe te kennen van ALS-puntenwolken, is van belang bij het genereren van deze 3D-producten die meerdere categorieën hebben en om gedetailleerde objectgeometrie vragen. Gemotiveerd door de topprestaties van deep learning algoritmen bij het begrijpen van scenes, onderzoekt dit proefschrift de semantische segmentatie van ALS-puntenwolken op basis van deep learning-algoritmen. We onderzoeken eerst hoe we representatieve kenmerken kunnen leren uit ALS-puntenwolken (hoofdstuk 2). Vervolgens richten we ons op hoe we de handmatige annotatie-inspanningen kunnen verminderen om een deep learning model voor semantische segmentatie te trainen. We onderzoeken actief leren (hoofdstuk 3) om informatieve punten te selecteren en te annoteren, en zwakke supervisie (hoofdstuk 4) om alleen zwakke labels te annoteren voor de puntsgewijze voorspellingstaak.

Om de deep learning netwerken in staat te stellen representatieve kenmerken te leren uit ALS gegevens en verschillende niveaus van naburige informatie te betrekken om puntsgewijze geometrische kenmerken te extraheren, ontwerpen we eerst een lokale feature-extractor en verkennen vervolgens contextuele informatie op zowel object als globaal niveau. Het voorgestelde LGEnet gebruikt zowel 2D- als 3D-convoluties als lokale encoders. De combinatie van 2D- en 3D-convoluties stelt het netwerk in staat om meer discriminerende kenmerken te leren voor langwerpige objecten die verdeeld zijn over horizontale vlakken. Bovendien wordt contextuele informatie op objectniveau onderzocht via de voorgestelde op segmenten gebaseerde Edge Conditioned Convolution (SegECC), waarbij grafieken worden geconstrueerd tussen segmenten. Vervolgens wordt aan het einde van het netwerk een ruimtelijk kanaalattentie module geplaatst. Deze ruimtelijke aandacht module modelleert de globale onderlinge afhankelijkheden tussen punten door de paarsgewijze correlaties tussen alle punten binnen een invoerbol te berekenen. De kanaalattentie module schat de overeenkomsten tussen kanalen, met als doel het leren van klassespecifieke discriminerende kenmerken te verbeteren.

Om de vereiste annotatie-inspanningen voor de training van deep learning-modellen te verminderen, stellen we een actief en incrementeel leerraamwerk voor semantische segmentatie van ALS-

puntenwolken voor. In dit raamwerk selecteren we iteratief puntenwolktegels uit de niet-gelabelde pool en verrijken vervolgens stapsgewijs de modelkennis. Voor de selectiecriteria implementeren we twee gegevensafhankelijke onzekerheidsmetrieken (puntentropie en segmententropie) en één modelafhankelijke metriek (wederzijdse informatie). De door ons voorgestelde segment entropie schat de semantische heterogeniteit binnen geometrische homogene eenheden en bereikt de beste prestaties voor de ALS-datasets. In plaats van voor elke iteratie vanaf nul te trainen, finetunen we het vorige netwerk op de vergrote gelabelde dataset en dit bespaart aanzienlijk op de trainingstijd.

We onderzoeken ook hoe we de annotatie-inspanningen voor de deep learning-training kunnen verlichten door zwakke subcloud-labels te gebruiken in plaats van puntsgewijze ground truth voor ALS-datasets. De eerste stap is het trainen van een classificatienetwerk met zwakke subcloud-labels, waarna de puntsgewijze pseudo-labels op de trainingsgegevens worden geproduceerd door het getrainde classificatienetwerk. De tweede stap is het exploiteren van de geproduceerde pseudo-labels om een segmentatienetwerk te trainen dat vervolgens voorspellingen maakt over de testgegevens. De prestaties van het classificatienetwerk worden versterkt door een verlies van overlapgebieden en een hoogte-attentie. Het verlies van het overlapgebied zorgt voor meer lokalisatieaanwijzingen voor het classificatienetwerk en de hoogte-aandacht stelt het classificatienetwerk in staat om meer representatieve kenmerken van ALS-gegevens te leren. Voor de training van het segmentatienetwerk gebruiken we een gesuperviseerd contrastief verlies dat de onderliggende correlaties van klassespecifieke kenmerken blootlegt. Dit verlies stelt het segmentatienetwerk in staat om op effectieve wijze meer representatieve klassespecifieke kenmerken te leren van onnauwkeurige puntsgewijze pseudo-labels. Het voordeel is maximaal wanneer het netwerk dynamisch de labels bijwerkt voor die punten waarvan de door het classificatienetwerk gegenereerde pseudo-labels niet betrouwbaar zijn.

Concluderend onderzoekt dit proefschrift deep learning-algoritmen voor semantische segmentatie van ALS-puntenwolken. Eén netwerk wordt voorgesteld om representatieve kenmerken uit ALS-gegevens te extraheren en er worden twee methoden worden voorgesteld om de annotatie-inspanningen voor de training van semantische segmentatienetwerken te verminderen. In de toekomst zou verder onderzocht kunnen worden hoe zelf-attentie mechanismen als feature-extractors kunnen worden gebruikt. Ook zijn pre-training zonder toezicht en domeinaanpassing mogelijke oplossingen om de vereiste

annotatie-inspanningen voor de training van semantische segmentatienetwerken te verminderen.

Author's Biography

Yaping Lin was born on the 12th of October 1994 in Puyang, Henan, China. She received her Bachelor's degree in the University of Nottingham Ningbo China in 2016 and graduated with a first class degree. Then, with the Holland scholarship, she obtained her Master's degree in the Faculty of Geo-information Science and Earth Observation (ITC), University of Twente, the Netherlands in 2018. After her master's study, she stayed at ITC and started her Ph.D research from the 18th of October 2018 with the China Scholarship Council (CSC) scholarship. This thesis is the output of her research. She published several conference and journal papers. She authored and co-authored the following publications:

Lin, Y., Vosselman, G., Yang, M.Y., 2022. Weakly supervised semantic segmentation of airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 187, 79–100.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2021. Local and global encoder network for semantic segmentation of Airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 176, 151–168.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2020a. Active and incremental learning for semantic ALS point cloud segmentation. ISPRS Journal of Photogrammetry and Remote Sensing 169, 73–92.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2020b. Efficient Training of Semantic Point Cloud Segmentation via Active Learning, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 243–250.

Lin, Y., Nex, F., Yang, M.Y., 2019. Semantic façade segmentation from airborne oblique images. Photogrammetric Engineering and Remote Sensing 85, 425–433.

Lin, Y., Yang, M.Y., Nex, F., 2018. Semantic Building Façade Segmentation from Airborne Oblique Images, in: ISPRS TC II Mid-Term Symposium

Yang, Z., Jiang, W., **Lin, Y.**, Elberink, S.O., 2020. Using training samples retrieved from a topographic map and unsupervised segmentation for the classification of airborne laser scanning data. Remote Sensing 12, 1–18.

Huang, S., Nex, F., **Lin, Y.**, Yang, M.Y., 2019. Semantic Segmentation of Building in Airborne Images, in: International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences.

ITC Dissertation List

https://www.itc.nl/Pub/research_programme/Research-review-and-output/PhD-Graduates