Deep Learning for 3D Information Extraction from Indoor and Outdoor Point Clouds

by

Ying Li

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Geography

Waterloo, Ontario, Canada, 2020

Examining Committee Membership

The following served on the Examining Committee members for this thesis. The final decision of the Examining Committee is by majority vote.

Supervisor:	**Dr. Jonathan Li** Geography & Environmental Management, University of Waterloo
Co-Supervisor:	**Dr. Dongpu Cao** Mechanical and Mechatronics Engineering, University of Waterloo
Internal Member:	**Dr. Richard Kelly** Geography & Environmental Management, University of Waterloo
Internal Member:	**Dr. Michael A. Chapman** Civil Engineering (Adjunct GEM), Ryerson University
Internal-External Member:	**Dr. Linlin Xu** Systems Design Engineering, University of Waterloo
External Member:	**Dr. Derek D. Lichti** Geomatics Engineering, University of Calgary

AUTHOR'S DECLARATION

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

This doctoral thesis is accomplished in the manuscript option, following the graduation guidelines administered by the joint Waterloo-Laurier Graduate Program in Geography. Three required manuscripts have been published as refereed journal papers. I was the co-author with major contributions on designing the methods, implementation and writing the papers. My supervisors Prof. Jonathan Li and Prof. Dongpu Cao are the corresponding authors and dominants my efforts. Other co-authors also contributed to the implementation and proofing of these works:

Li, Y., Ma, L., Zhong, Z., Cao, D. and Li, J. 2020. TGNet: Geometric Graph CNN on 3D Point Cloud Segmentation. IEEE Transactions on Geoscience and Remote Sensing, vol. 58, no. 5, pp. 3588-3600.

This paper is incorporated in Chapter 2 and Chapter 3 of this thesis.

Li, Y., Ma, L., Tan, W., Sun, C., Cao, D., Li, J. 2020. GRNet: Geometric relation network for 3D object detection from point clouds, ISPRS Journal of Photogrammetry and Remote Sensing, 165, 43-53.

This paper is incorporated in Chapter 2 and Chapter 4 of this thesis.

Li, Y., Ma, L., Tan, W., Sun, C., Cao, D., Li, J. 2020. 2020. 2D-Driven 3D Object Detection from Point Clouds, IEEE Transactions on Intelligent Transportation Systems, submitted.

This paper is incorporated in Chapter 2 and Chapter 5 of this thesis.

Li, Y., Ma, L.**,** Zhong, Z., Liu, F., Cao, D. and Li, J. 2020. Deep learning for LiDAR point clouds in autonomous driving: A review. IEEE Transactions on Neural Networks and Learning Systems, doi:10.1109/TNNLS.2020.3015992.

This paper is partially incorporated in Chapter 1 and Chapter 6 of this thesis.

Abstract

This thesis focuses the challenges and opportunities that come with deep learning in the extraction of 3D information from point clouds. To achieve this, 3D information such as point-based or object-based attributes needs to be extracted from highly-accurate and information-rich 3D data, which are commonly collected by LiDAR or RGB-D cameras from real-world environments. Driven by the breakthroughs brought by deep learning techniques and the accessibility of reliable 3D datasets, 3D deep learning frameworks have been investigated with a string of empirical successes. However, two main challenges lead to the complexity of deep learning based per-point labeling and object detection in real scenes. First, the variation of sensing conditions and unconstrained environments result in unevenly distributed point clouds with various geometric patterns and incomplete shapes. Second, the irregular data format and the requirements for both accurate and efficient algorithm pose problems for deep learning models.

To deal with the above two challenges, this doctoral dissertation mainly considers the following four features when constructing 3D deep models for point-based or object-based information extraction: (1) the exploration of geometric correlations between local points when defining convolution kernels, (2) the hierarchical local and global feature learning within an end-to-end trainable framework, (3) the relation feature learning from nearby objects, and (4) 2D image leveraging for 3D object detection from point clouds. Correspondingly, this PhD thesis proposes a set of deep learning frameworks to deal with the 3D information extraction specific for scene segmentation and object detection from indoor and outdoor point clouds.

Firstly, an end-to-end geometric graph convolution architecture on the graph representation of a point cloud is proposed for semantic scene segmentation. Secondly, a 3D proposal-based object detection framework is constructed to extract the geometric information of objects and relation features among proposals for bounding box reasoning. Thirdly, a 2D-driven approach is proposed to detect 3D objects from point clouds in indoor and outdoor scenes. Both semantic features from 2D images and the context information in 3D space are explicitly exploited to enhance the 3D detection performance. Qualitative and quantitative experiments compared

with existing state-of-the-art models on indoor and outdoor datasets demonstrate the effectiveness of the proposed frameworks. A list of remaining challenges and future research issues that help to advance the development of deep learning approaches for the extraction of 3D information from point clouds are addressed in the end of this thesis.

Acknowledgements

First of all, I would like to express my deep gratitude to my supervisor, Prof. Jonathan Li for his insightful advice and great help for guiding me to overcome academic challenges and achieve several successes in my doctoral career. He taught me how to deal with challenging academic issues, how to conduct research work, how to write journal papers, and how to be a reviewer. The knowledge I learned during this period will benefit me for life. I believe we will have more promising and professional cooperation in the future.

Then, I also would like to express my sincerely gratitude to my co-supervisor, Prof. Dongpu Cao for his full support and constructive suggestions for my academic study. Many thanks for his constant help, positive remarks, and intellectual guidance from my first academic journal paper to the end of this thesis. I remember what he told me "when you graduate, you need to face the world", which gives my courage to face future challenges.

In addition, many thanks to my colleagues and friends Lingfei Ma, Weikai Tan, Dedong Zhang, and Zilong Zhong in Geospatial Sensing and Data Intelligence Lab at the University of Waterloo, for establishing a collaborative and efficient environment and always being ready to help. Thanks to colleagues Dr. Huilong Yu, Chen Sun, and Yaodong Cui in CogDrive Lab at the University of Waterloo for their collaborative support, constructive discussion, and teamwork. I would like to thank my friends Ming Liu, Ruijin Sun, Minghao Liu, Qi Liu, Prof. Yan Liu, Shuping, Yajun Zheng, Zhiwei Shang, Weiya Ye, Yue Gu, Mengge Chen, Liyuan Qing et al., for bringing me happiness and helping me balance the academic and life.

Besides, I would like to thank my thesis committee members Prof. Richard Kelly from the Department of Geography and Environmental Management and Prof. Linlin Xu from the Department of Systems Design Engineering at the University of Waterloo, Prof. Michael Chapman from the Ryerson University, and Prof. Derek Lichti from the University of Calgary for their time and commitment. Many thanks to Mr. Mike Lackner from the Centre of Mapping, Analysis & Design (MAD), Faculty of Environment, University of Waterloo, for his technical

supports. Besides, I would like to thank Dr. Dunn Emma at the University of Waterloo Writing and Communication Centre for carefully proofreading with my journal paper and this thesis.

Most importantly, I would like to thank my family, including my parents, my brother, my boyfriend for their incredible love and unreserved supports throughout these years.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

2D	Two-dimensional
3D	Three-dimensional
3DMV	3D Multi-View
3D-SIS	3D Semantic Instance Segmentation
Adam	Adaptive Moment Estimation
AP	Average Precision
AVOD	Aggregate View Object Detection
BEV	Birds Eye View
CNN	Convolutional Neural Network
COG	Clouds of Oriented Gradients
ContFuse	Continuous Fusion
CRF	Conditional Random Field
CRFasRNN	Conditional Random Fields as Recurrent Neural Networks
DGCNN	Dynamic Graph CNN
DMI	Distance Measurement Indicator
DSS	Deep Sliding Shapes
FN	False Negatives
FP	False Positives
FP	Feature Propagation
FPS	Farthest Point Sampling
GAC	Graph Attention Convolution

GACNet	Graph Attention Convolution Network
GAN	Generative Adversarial Network
GCN	Graph Convolutional Networks
GeoConv	Geometric Convolution
GMM	Gaussian Mixture Model
GNSS	Global Navigation Satellite System
GPU	Graphics Processing Units
GRNet	Geometric Relation Network
HD	High-definition
IMU	Initial Measuring Unit
IoU	Intersection over Union
IPOD	Intensive Point-based Object Detector
KNN	K-Nearest Neighbour
LiDAR	Light Detection and Ranging
mAP	Mean Average Precision
mIoU	Mean Intersection over Union
MLP	Multilayer Perception
MLS	Mobile Laser Scanning
MoNet	Mixture Model Network
MV3D	Multi-View 3D
MVCNN	Multi-View CNN
NMS	Non-Maximum Suppression

OA	Overall Accuracy
POS	Position and Orientation System
ReLU	Rectified Linear Unit
RPN	Region Proposal Network
S3DIS	Stanford Large Scale 3D Indoor Spaces
SA	Set-Abstraction
SLAM	Simultaneous Localization and Mapping
SVM	Support Vector Machine
TGConv	Taylor Gaussian Convolution
TGNet	Taylor Gaussian Mixture Model
TN	True Negatives
TP	True Positives

Chapter 1

Introduction

This chapter introduces the background and motivation of the use of deep learning for extraction of 3D information from point clouds and the organization of this thesis. Sections 1.1 introduces the background and challenges of extraction of 3D information from point clouds. Section 1.2 describes the motivation of the deep learning based 3D information extraction frameworks specific for semantic segmentation and object detection. Section 1.3 presents the objective of this study. The overall structure of this thesis is described in Section 1.4.

1.1 Background

Nowadays, the development of 3D remote-sensing technology facilitates the collection of indoor and outdoor 3D data in a faster, safer way with high accuracy, which significantly upgrades the results of perception, modeling, and survey for road infrastructures and indoor environments (Li et al., 2016). These applications can be concluded in two main aspects: (1) real-time environment perception and processing for scene understanding and object detection (Yang et al., 2018); (2) high-definition (HD) map and urban model generation and construction for reliable localization and referencing (Levinson et al., 2011). To provide accurate products or outputs for these applications, efficient and effective 3D information extraction is of great importance.

3D information can be extracted from images (Li et al., 2019) or point clouds (Li, 2017). Although images captured by digital camera can provide color, texture, and semantic information for objects with low cost and high efficiency, they lack 3D geo-referenced information (Ma et al., 2018). In addition, the presence of partial or fully distortion, occlusion, and truncation in images affect the 3D information extraction performance. A Point cloud, with 3D topological and geo-referenced information, is a set of data points in space and it can provide more accurate 3D pose and position information for objects. Each point has its set of X, Y and Z coordinates. Compared with image-derived height data, point cloud data have fewer

occlusion and truncation problems. Thus, our study focusses on 3D information extraction from point clouds.

Point cloud data are commonly acquired by either a laser scanner or an RGB-D depth camera. Normally one or two laser scanners are mounted on a car or a van together with an integrated GNSS/IMU (Global Navigation Satellite System and Initial Measuring Unit), a POS (position and orientation system) subsystem, optical cameras, and a distance measurement indicator (DMI) device to form a so-called mobile laser scanning (MLS) system or a mobile LiDAR system (Ma et al., 2018). Such an MLS system has been used to collect 3D point clouds covering large-scale complex roadway environments (Guan et al., 2016; Ma et al., 2018). Because the GNSS signals are not available in most indoor or underground areas, named GNSS-denied environments, the approach to simultaneous localization and mapping (SLAM) has been developed for indoor mapping (Dissanayake et al., 2001).

The RGB-D camera can be used to capture both a colored (RGB) image and perform a depth (D) measurement. RGB-D data can be lifted to point clouds within the known camera matrix and depth information (Qi et al., 2018). Compared with MLS, the RGB-D camera is easier to operate and has lower cost. However, it is sensitive to illumination change, occlusion, and truncation, and not suitable for long distance sensing. Thus, the RGB-D camera is commonly applied in short-distance 3D sensing such as indoor environment scanning.

To provide highly-accurate and geo-referenced data for 3D information extraction in different scenarios, MLS system usually scans the road continually for dozens or even hundreds of kilometers (Geiger et al., 2013), while RGB-D camera scans hundreds of rooms in indoor buildings (Song et al., 2015). Consequently, large-scale 3D point clouds can be produced.

However, how to process the massive inhomogeneous and unstructured point clouds is critical to 3D information extraction. The variation of ranging and imaging conditions and the complexity of environments result in significant variations for objects in point cloud data. Thus, there are several challenges when processing point cloud data:

- **Diversified point density and reflective intensity**. Due to the scanning mode of laser scanners, the density and intensity for objects vary considerably. The distribution of these two characteristics highly depends on the distance between objects and laser scanners (Wang et al., 2015; Hackel et al., 2017; Wen et al., 2019). For example, objects that are far away from the sensors have low point density, but objects that are close to the sensors have high point density. Besides, the ability of scanning sensors, the time constraints of scanning, and the needed resolutions also affect the point cloud distribution and intensity.
- **Incompleteness**. Objects that are represented by point clouds are commonly incomplete (Tagliasacchi et al., 2009). This mainly results from the occlusion caused by objects (Guan and Neumann, 2016), the cluttered background in urban scenes (Wang et al., 2015; Kumar et al., 2019), and the unsatisfactory material surface reflectivity. Such problems are severe in real-time capturing of moving objects, which result in large gaping holes and severe under-sampling.

Therefore, there is an urgent need to develop intelligent algorithms and methods for extracting the target-based 3D geometric information from point clouds. Besides, data representation for point clouds and algorithm requirements for both accuracy and efficiency should also be considered when processing point clouds. Commonly, the performance of these models is measured by the accuracy, precision, recall, etc. The training time or inference time within the same experimental settings, the memory usage, and the model size are referenced as the efficiency evaluation metrics.

Thus, the overriding research question of this thesis is: what is the best way to infer information from a large amount of 3D point clouds? Traditionally, to extract accurate 3D information, the collected point clouds are processed step-by-step to acquire the desired target information (Guan et al., 2018). Commonly, the foreground points are segmented first from the raw input points to reduce the noise and background disturbance (Yu et al., 2015). Then, clustering methods are applied to cluster the foreground points into different individual parts. Finally, the point or object based information are extracted from these clusters (Yu et al., 2016),

e.g., semantic class, bounding box size, orientation, and geometric shape. Although these traditional methods have been applied in many cases, they suffer the following two disadvantages:

- **Low generalization**. Hand-designed features are commonly proposed for a target task, which cannot quantitatively generalize well to other tasks (Zhong, 2019).
- **Semi-automatic**: These type of methods extract the target information with several steps, e.g., feature design, manual parameter selection, and coarse-to-fine clustering. Such semi-automatic operation cannot meet the requirement for real-time perception and localization.

In recent years, deep learning methods utilize multiple layers to progressively learn high-level features from the input data. With the advancement of hardware techniques such as faster Graphics Processing Units (GPUs), faster network connectivity, and the appearance of reliable public 3D datasets, deep learning methods applied to 3D scene segmentation, object detection and classification have emerged and achieved noticeable increased performances in accuracy and efficiency (Qi et al., 2017; Li et al., 2018; Zhang and Zhang, 2018). One reason is that the feature design is omitted and not required, which relieves more chance for model itself to fully exploit all potential features. In addition, the multi-task training for different applications can be achieved simultaneously with multi-layer neural networks (Liang et al., 2019).

Consequently, the objective of this thesis is to design a set of deep learning frameworks to extract 3D information from massive and irregular point clouds in different scenes and achieve better results in accuracy and efficiency compared with state-of-the-art 3D deep models (Qi et al., 2017; Li et al., 2018; Zhang and Zhang, 2018; Benshabat et al., 2018; Shi et al., 2019; Wang et al., 2019) in several cases.

1.2 Motivation

2D convolutional neural networks (CNNs) have been developed rapidly in recent years for the discriminate feature learning and high generalization capability (Lecun et al., 2015). Specifically, CNN is a type of neural networks that apply the convolution, a mathematical

operation, in place of general matrix multiplication in at least one of neural layers. However, the irregular and unstructured data format of point clouds poses a great challenge for traditional 2D CNN models. Conventionally, these models are mainly applied to data with a regular structure, such as the 2D pixel array (Long et al., 2015). Thus, in order to apply CNNs to irregular 3D point cloud data, Multi-view CNN (MVCNN) (Su et al., 2015) is proposed as the pioneer in exploiting 2D deep models to learn 3D information. Multiple views of interest objects or scenes are captured from different orientations and then input to 2D CNNs for object class prediction. MVCNN-MultiRes (Qi et al., 2016), RotationNet (Kanezaki et al., 2018), and 3D Multi-View (3DMV) (Dai and Niesner, 2018) further improve the 3D information extraction performance by considering multiple resolution features and oriented-view cues. View-based 3D models can exploit established 2D deep architectures and datasets, however, the projection from 3D space to 2D views can lose some geometrically-related spatial information in 3D space.

To explore the 3D geometric attributes of point clouds, 3D ShapeNet (Wu et al., 2015) is proposed to apply CNNs to volumetric data, where point clouds are divided into regular grids with certain size to describe the distribution of data in 3D space. A more advanced voxel-based data representation of a point cloud is the octree-based grids (Riegler et al., 2017; Tatarchenko et al., 2017), which use adaptive size to divide the 3D point cloud into cubes. However, the computation cost increases cubically with the increment of input data size or resolution, which limit the model's performance in large-scale or dense point clouds.

Voxel grids and view images are Euclidean-structured data which are suitable for using traditional 2D convolutional operation to extract distinctive spatial features such as edges and key-points (Li et al., 2020). But they are constrained by the local receptive fields as they scan the space with fixed strides. Besides, the original geospatial information in 3D space cannot be well-kept during point cloud projection or voxelization. For example, the depth information along Z axes will be lost when transforming point clouds from 3D scene to (x, y) in 2D image dimension.

Although point clouds can preserve the original 3D geospatial information in 3D scenes, the unstructured and irregular data format limits the application of conventional 2D CNNs. After Qi et al. (2017) proposed the first point cloud based deep model (PointNet), which takes the point clouds directly as input, many deep learning models are later designed on this basis to extract the geospatial structure features of a point cloud, such as PointNet++ (Qi et al., 2017) and PointCNN (Shi et al., 2019). These methods facilitate the development of deep learning in 3D geospatial information extraction tasks with robust and efficient performances.

Graphs as a type of non-Euclidean data structure can also be used to represent point cloud data. Each graph node corresponds to a point and the edges represent the relationship between each point neighbours (Yi et al., 2017; Simonovsky and Komodakis, 2017). Those graph CNNs define convolutions directly on the graph in the spectral and non-spectral (spatial) domain, operating on groups of spatially close neighbours (Benshabat et al., 2018; Wang et al., 2019; Wang et al., 2019). The advantage of graph-based models is that the geometric relationships among points and their neighbours are exploited. Thus, spatially-local correlation features are extracted from the grouped edge relationships. Figure 1.1 introduces the chronological overview of 3D deep learning networks since 2015 and four data representation examples of point cloud data. Multi-view and voxel representations are firstly developed in 2015. With the publication of PointNet (Qi et al., 2017) and ECC (Simonovsky and Komodakis, 2017) in 2017, point-based and graph-based representations are then explored in 3D information extraction.

Although the multi-view and voxel grid data formats can leverage existing mature 2D CNNs, point cloud and graph representation of a point cloud can preserve the raw 3D geospatial information in 3D space and the internal local structure of objects. Thus, this thesis mainly focuses on the point cloud and graph representation of point cloud data when developing deep learning models for 3D information extraction.

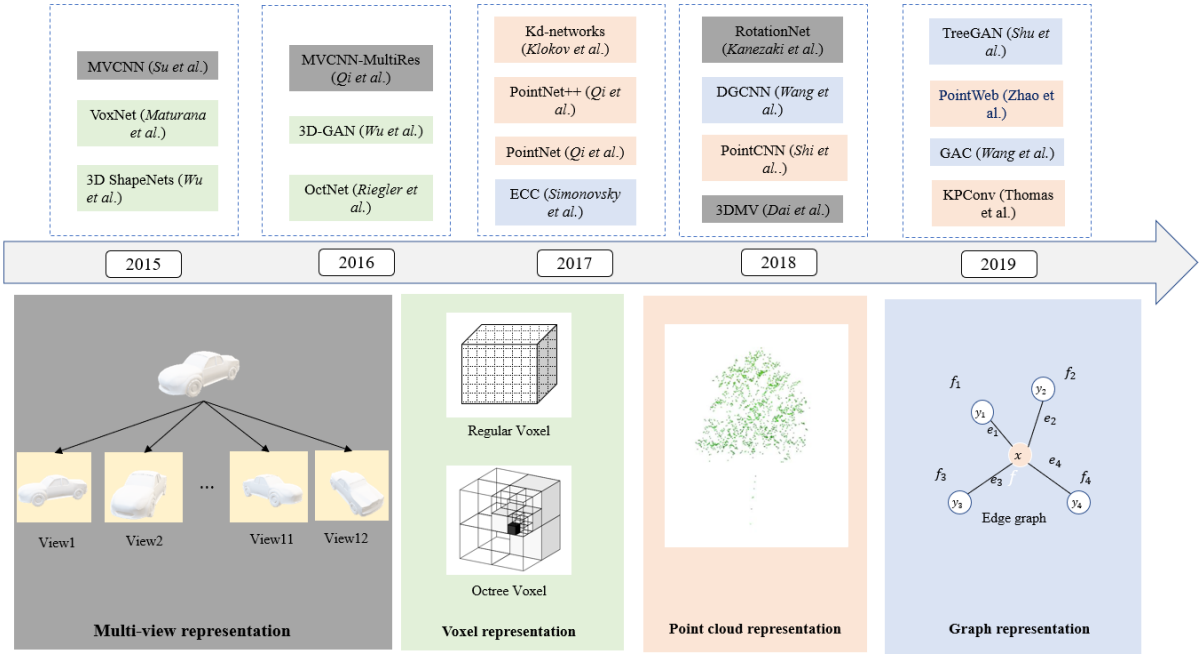

Figure 1.1: Chronological overview of 3D deep learning networks.

The information extracted from point clouds can be classified as point-based and object-based information. The related tasks of such information can be roughly divided into three types: 3D point cloud segmentation (Armeni and Zamir, 2016), 3D object detection (Luo et al., 2019), and 3D object classification (Gao et al., 2018). Scene segmentation focuses on the per-point label prediction, while detection and classification concentrate on integrated point set or object points labeling. The semantic information for each point can attribute to foreground point extraction in object detection and classification. For object detection, the classification module is generally enrolled to predict the detected object semantic label. Thus, in this thesis, point-based semantic segmentation and object-based detection tasks are performed to extract 3D information in indoor and outdoor scenes.

When applying deep learning on these tasks, point-exact features and object-based patch features are required (De Brabandere et al., 2017). The primary problems and corresponding requirements for point-based semantic segmentation and object-based detection are:

1) Geometric patterns of objects vary enormously (Ren and Sudderth, 2018). The designed CNNs should consider the geometric variation.

2) Most 3D object shapes are incomplete. The proposed deep models can predict the semantic label or oriented bounding box with missing information.

3) The exploration of local geometric correlations between the input and its neighbouring coordinates or features is hard to achieve. The proposed CNNs can learn such geometric information.

4) The target objects only occupy a very limited amount of the whole input data. The proposed framework can extract targets accurately from backgrounds.

5) Sensors only capture surfaces of objects, 3D object centres are likely to be in empty space, far away from any point (Qi et al., 2019). Thus, the constructed framework can collect sufficient object information around the object centre.

These challenges lead to the complexity of per-point labeling and object localization and detection in real-world environments.

Commonly, MLS and RGB-D cameras can provide corresponding images for the acquired point clouds. Objects in large-scale and sparse point clouds are hard to localize accurately. How to reduce searching area when detecting objects is a key challenge. Some methods leverage 2D images to reduce 3D searching space for object localization, e.g., F-PointNet (Qi et al., 2018; Wang and Jia, 2019), where the 3D detector extract the amodal bounding box in a 3D frustum space which is lifted from a 2D proposal in the image. Several papers (Chen et al., 2017; Ku et al., 2018) reduce the searching work by projecting 3D points to 2D images, and then use proposal-based network for object detection in 2D images. However, how to exploit 2D images to leverage 3D detection remains an open problem.

1.3 Objectives of the Study

To handle the above mentioned problems in semantic segmentation and object detection and explore the 2D images to assist 3D object detection, this thesis proposes a set of deep learning frameworks to extract point-exact or object-based information from point clouds. Both complex road scenes and indoor scenes are covered. The objectives of these proposed algorithms are to achieve higher accuracy and robustness, but less computational time than the state-of-the-art methods. The specific objectives of this thesis can be described as follows:

The first objective is to address the 3D point cloud segmentation problem in both indoor and outdoor environments by considering the geometric relationships for each point with its local neighbours. An end-to-end encoder-decoder structure with a CRF refinement layer is constructed to predict the per-point label with efficiency and accuracy.

The second objective is to propose an end-to-end point cloud geometric relation network focused on 3D object detection. Intra-object geometric features and inter-object relation features are learned and used to enhance the 3D object detection performance in an end-to-end trainable way.

The third objective is to propose a 2D-driven 3D object detection architecture, which can exploit the 2D images to assist 3D object detection. The geometric features learned from the point clouds and semantic features generated from the corresponding single image are leveraged in bounding box reasoning.

1.4 Structure of the Thesis

This doctoral dissertation aims at proposing a set of deep learning based 3D information extraction algorithms from point clouds in indoor and outdoor scenes with robust and efficient performances. Figure 1.1 illustrates the overall structure of this thesis. The corresponding arrangement of this thesis is shown as follows:

Chapter 2 reviews the basic knowledge of CNN and point cloud convolution, a variety of existing deep learning studies related to point cloud segmentation, 3D object detection, and sensor fusion for 3D object detection. The indoor and outdoor datasets that can be used to train 3D deep models for segmentation and detection tasks are provided. To evaluate the algorithm performance in accuracy and efficiency and conduct comparison with existing state-of-the-art methods, several evaluation metrics for segmentation and detection are also introduced.

Chapter 3 details a geometric graph convolution architecture for per-point semantic labeling in indoor and outdoor scenes, which explore the geometric attributes among local points to improve the segmentation results.

Chapter 4 introduces a geometric relation framework for 3D object detection from point clouds. The intra-object geometric features and inter-object relation features are prompt to enhance the 3D detection performance.

Chapter 5 proposes the 2D-driven 3D object detection framework to leverage 2D images for 3D object detection. Semantic cues from 2D detection results and context features learned from 3D detectors are fused to boost 3D detection accuracy in indoor and outdoor scenes.

Chapter 6 concludes this research with a summary of contributions and details future research directions.

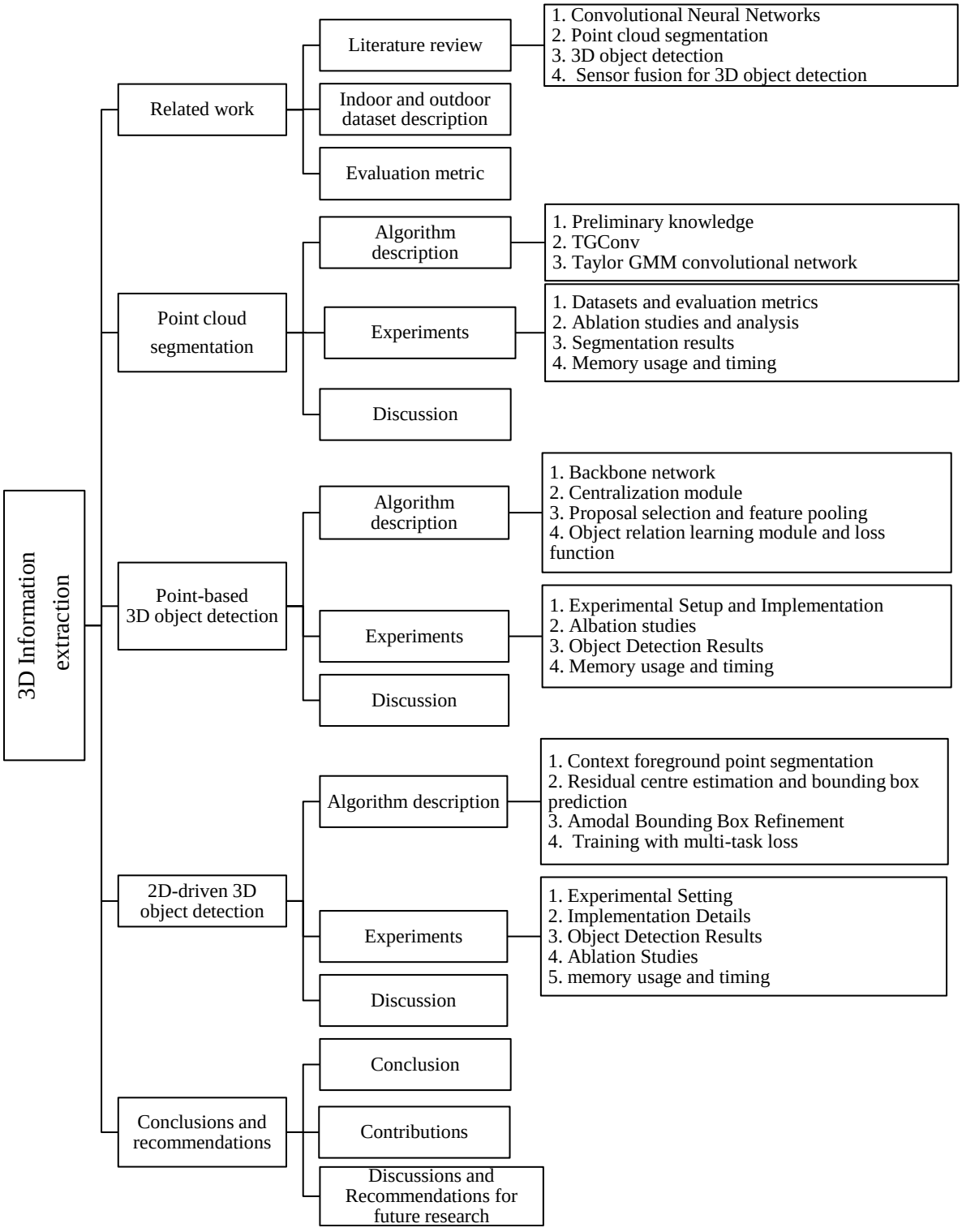

Figure 1.2: Framework of this thesis.

Chapter 2

Related Work

This chapter briefly reviews the related work in the scope of deep learning-based 3D information extraction from LiDAR and RGB-D data. Section 2.1 provides the basic knowledge of CNN and point cloud convolution. Section 2.2 provides the problem definition of point cloud segmentation and object detection tasks, and related studies of deep learning methods in point cloud segmentation and 3D object detection. Sensor fusion 3D object detection methods are also reviewed. Section 2.3 presents the related evaluation metrics for accuracy and efficiency. Section 2.4 concludes this chapter.

2.1 Point Cloud Convolution

2.1.1 Convolutional Neural Networks

The convolutional neural network (CNN) is one of the most popular deep learning algorithms. CNNs are featured with shift-invariant based on their weight sharing architecture and translation invariance characteristics. Commonly, a CNN is composed of an input and an output layer, and multiple hidden layers (Lecun et al., 2015). The input layer mainly pre-processes the input data. The hidden layers consist of a series of neural layers, such as the convolutional layer, the activation layer, the pooling layer, and the fully connected layer. The convolutional layer, which convolves with a multiplication or other dot product, is used to extract features from the input data. Each element of the convolutional kernel corresponds to a weight coefficient and a bias vector. The activation layer refers to the non-linear mapping of the output feature map of the convolutional layer. The pooling layer is sandwiched between successive convolutional layers to reduce the spatial size of the output feature maps to compress the number of parameters and hence to control overfitting. Neurons in a fully connected layer have full connections to all activations in the previous layer. Their activations can be computed with a matrix multiplication followed by a bias offset.

The convolution and nonlinearity on 2D grid data (e.g., images) can be expressed as follows (Liu et al., 2020):

$$x_j^l = \sigma\left(\sum_{i=1}^{N^{l-1}} x_i^{l-1} * w_{i,j}^l + b_j^l\right) \tag{2.1}$$

where x_i^{l-1} represents the ith input feature map at layer $l-1$, x_j^l denotes the jth output feature map at layer l. * is the convolution operation. $w_{i,j}^l$ and b_j^l represent the weights and bias at layer l. N^{l-1} is the number of feature maps at layer l-1. The $\sigma(\cdot)$ is the elementwise nonlinear function.

2.1.2 Point Cloud Convolution

Compared to kernels defined on 2D grid structures, designing convolutional kernels for 3D point clouds is hard to achieve. In order to extract discriminate point features from irregular point clouds, the modification of standard convolution is conducted.

Similar to 2D kernels, the 3D point convolution defines a set of spatial filters applied locally in the point cloud. Given the points x_i from $\mathcal{P} \in R^{N\times 3}$ and their corresponding features f_i from $\mathcal{F} \in R^{N\times D}$, the general point convolution of $\mathcal{F}$ by a kernel g at a point $x \in R^3$ is defined as:

$$(\mathcal{F} * g)(x) = \sum_{x_i \in \mathcal{N}_x} g(x_i - x)\, f_i \tag{2.2}$$

where $\mathcal{N}_x = \{x_i \in \mathcal{P} | \|x_i - x\| \le r\}$ is denoted as the neighbor set of point x, $r \in R$ is the selected radius. The neighbor point set is commonly searched using Ball query or K nearest neighbor (KNN) search (Qi et al., 2017).

However, there are two ways to define convolutional kernels. The first one defines convolutional kernels on a continuous space, where the weights for neighboring points are related to the spatial distribution with respect to the center point (Guo et al., 2020). The second one defines convolutional kernels on regular grids, where the weights for neighboring points are related to the offsets with respect to the center point (Guo et al., 2020).

2.2 Deep Learning-based 3D Information Extraction Techniques: An Overview

2.2.1 Point Cloud Segmentation

Problem definition: Point cloud segmentation is the process to cluster the input data into several homogeneous regions, where points in the same region have identical attributes (Nguyen and Le, 2013). Each input point is predicted with a semantic label, such as ground, tree, building. The task can be summarized as: given a set of ordered 3D points $X = \{x_1, \cdots, x_n\}$ with $x_i \in R^3$ and a candidate label set $Y = \{y_1, \cdots, y_k\}$, assign each input point x_i with one of the k semantic labels (Huang et al., 2018). Segmentation results can further support object detection and classification.

Point cloud segmentation algorithms based on deep learning can be grouped into two main categories according to their data structures: Euclidean-structured data and non-Euclidean data (Ahmed et al., 2018). The Euclidean-structured data refer to the volumetric data structure which has gridded regular data structure, while the non-Euclidean data refer to the irregular and unstructured data formats such as point cloud and graphs.

Euclidean-structured data models. The Euclidean-structured data are suitable for convolutional operation to extract distinctive spatial features such as edges and key-points. Volumetric-based models (Wu et al., 2015; Klokov and Lempitsky, 2017; Riegler et al., 2017; Zhou and Tuzel, 2018) are the most representative frameworks in existing 3D Euclidean-structured deep models applied on large-scale point clouds. The inputs of these methods are 3D volumetric grids voxelized from the raw point clouds. In early voxel-based networks, convolution is operated in regular and uniform voxel grids (Wu et al., 2015). This leads to an excessive requirement of memory footprints and high computation cost. Thus, the input point clouds are reduced to low resolutions to decrease memory and computation costs. For example, 3D ShapeNets (Wu et al., 2015) inputted volumetric grids with size 30*30*30 into CNN architecture, the geometric 3D shape was represented by binary variables with a probabilistic distribution of a 3D voxel grid. Instead of limiting the size of the input volume, Kd-networks (Klokov and Lempitsky, 2017) adaptively divided the input data into hierarchical grids, which

further reduce the computation cost and memory. OctNet (Riegler et al., 2017) hierarchically splitting the 3D space into a set of unbalanced octrees based on the density of the data. Then a modified CNN was applied to such a hybrid grid-octree data structure. However, the geometric features especially the intrinsic characteristic of 3D shapes and surfaces are not exploited. Such intrinsic characteristic can help the model differentiate objects with different shapes to improve the segmentation accuracy.

Non-Euclidean data models: As for the non-Euclidean data models, point cloud based models and graph-based models have achieved compelling results in several 3D tasks, such as segmentation (Qi et al., 2017; Qi et al., 2017), classification (Klokov and Lempitsky, 2017; Wang et al., 2019).

Point cloud based models: Volumetric input of 3D point clouds is still computational and complex, a simpler network PointNet was proposed by Qi et al. (2017), which takes point cloud directly as input. Symmetry function was used to irregular points and the spatial transform network was exploited to improve the geometric invariance of the proposed network. Spatial features of each input point were learned through the network. Then, the learned features were assembled across the whole region of point clouds. The outstanding performance of PointNet has achieved in 3D objects classification and segmentation tasks. However, local structure feature is not considered, which constrains its ability to learn fine-grained features and generalize to complex scenes. To solve the above problems, PointNet++ was proposed later by Qi et al. (2017) to compensate the local feature extraction problem. This network was applied in raw input point clouds with various resolutions and assemble local features using a hierarchical architecture. PointCNN (Li et al., 2018) proposed the χ-Conv to assemble features in each local range and developed a hierarchical network architecture. However, these models have not exploited the high-level geometric correlations of local neighbours, which limits their semantic segmentation accuracy.

Graph-based models: Related works about convolution on graphs can be classified into spectral and non-spectral approaches. The spectral-based graph CNNs are analogous to the operation between the Fourier transforms and eigen-decomposition of the graph Laplacian

(Bruna et al., 2013). Yi et al. (2017) defined the signal of point clouds in the Euclidean domain by the metrics on the graph nodes and related the convolution operation to the scaling signals based on eigenvalues of graph Laplacian. However, such operation is linear and dependent on the eigenvectors of the graph Laplacian, meaning that it is domain-dependent. Besides, the spectral filtering was defined based on the whole input data, which results in high computation cost. Thus, Wang et al. (2018) carried out the graph convolution on local point set and applied a recursive clustering and pooling operation to aggregate information from spectral-close nodes. Spatial-based graph CNNs is commonly operated on groups of spatially close neighbours. In Simonovsky and Komodakis (2017), features from local neighbourhoods were filtered and aggregated. Besides, the edge information based on the graph signal in the spatial domain was also exploited in constructing the convolution filters. Wang et al. (2019) also constructed a local neighbourhood graph to learn the local geometric features. The EdgeConv was applied on the edges connecting neighbouring pairs of each point. Besides, the given fixed graph was dynamically updated to extract high level local spatial information. However, not all neighbours contribute equally. Wang et al. (2019) introduced an attention scheme in graph-based point cloud segmentation by assigning specific attentional weights to different neighbouring points. This operation can dynamically adapt the kernel to different objects with various structures.

2.2.2 3D Object Detection

Problem definition: Given an arbitrary point cloud data, the goal of 3D object detection is to detect and locate the instances of predefined categories (e.g., cars, pedestrians, and cyclists, and return their geometric 3D location, orientation, and semantic instance label (Qi et al., 2018). Such information can be represented coarsely using a 3D bounding box which is tightly bounding the detected object (Zhou and Tuzel, 2018; Qi et al., 2019). This box is commonly represented as $(x, y, z, h, w, l, \theta, nclass)$, where (x, y, z) denotes the object (bounding box) centre position, (h, w, l) represents the bounding box size with width, length and height, and is the object orientation. The orientation θ refers to the rigid transformation that aligns the detected object to its instance in the scene, which are the translations in each of

the of x, y, and z directions as well as a rotation about each of these three axes (Beltran et al., 2018; Kundu et al., 2018). $nclass$ represents the semantic label of this bounding box (object).

Existing point-based 3D object detection methods can be grouped into two main types: view-based and 3D based. View-based methods project 3D points into 2D views and leverage mature 2D detectors to extract objects, while point based directly detect 3D objects from point clouds.

View-based Methods. In order to exploit existing 2D CNNs, some approaches first project point clouds into 2D views and then apply 2D CNNs to detect and localize objects from images. In early work by Xiang et al. (2015), Chen et al. (2016) and Mousavian et al. (2017), point clouds were projected initially to the camera image plane, then RGB images and shape attributes or occlusion patterns were exploited to predict 3D bounding boxes. Li et al. (2016) and Deng et al. (2017) treated depth data as 2D maps and applied 2D CNN learners to detect objects in 2D images. Luo et al. (2019) proposed a detection framework via fusing multi-view representations of point clouds to extract high-level features. Wen et al. (2019) projected point clouds into a horizontal plane and used a modified U-net to extract road markings. MV3D (Chen et al., 2017) projected LiDAR point clouds to bird's eye view images first and then constructed a region proposal network (RPN) (Ren et al., 2015) for 3D bounding box prediction. However, these methods have sub-optimal performance for accuracy for small object detection (e.g., pedestrians and cyclists) and multiple clutter object detection in the vertical direction. Due to the sparsity of point clouds, the projection of point clouds to 2D image planes produces sparse 2D point maps and losses 3D geometric information.

3D-based methods. Compared with view-based detection methods and 3D object detection using 2D-3D features, 3D-based approaches focus more on utilizing geometric features from point clouds. In work by Song and Xiao (2014) and Wang and Posner (2015), support vector machine (SVM) was adopted to classify 3D objects using hand-designed geodesic features extracted from point clouds. Then the object was localized via a sliding window search. Engelcke et al. (2017) extended the work by Wang and Posner (2015) by using 3D CNN instead of SVM on 3D voxelized grids. Ren and Sudderth (2016) designed new

geometric features for 3D object detection. Song and Xiao (2016) converted the entire scene represented by point clouds into volumetric grids and applied 3D volumetric CNNs on object proposal for classification. The computation costs for these methods are usually high because 3D convolutions and 3D space searching in large areas cost expensively. More recently, deep networks on point clouds were adopted by Yi et al. (2019) and Shi et al. (2019) to exploit the sparsity of the data. Considering the scanned points lying on the surface of the objects and the empty object centre, Qi et al. (2019) proposed a deep Hough voting network to shift the surface point to the object centre. This method achieved high accuracy in bounding box centre prediction and box size estimation.

2.2.3 Sensor Fusion for 3D Object Detection

When the point clouds are collected by RGB-D cameras (Song et al., 2015) or the mobile laser scanning system (Geiger et al., 2013), the corresponding images are also existed. Thus, to leverage the 2D imagery for 3D object detection, fusion-based approaches (Chen et al., 2017; Lahoud and Ghanem, 2017; Ku et al., 2018; Qi et al., 2018; Hou et al., 2019; Qi et al., 2020) have been developed rapidly and achieved a notable success.

There are two type methods for fusing 2D and 3D sensing data (Qi et al., 2020): 2D-driven and 2D-3D feature fusion. 2D-driven strategies (Lahoud and Ghanem, 2017; Qi et al., 2018) first extract objects from 2D images, which are then back-projected to 3D space to guide the search area. 2D features such as color and semantic information are exploited for 3D object detection. These methods can leverage the mature 2D detectors for object detection from images and reduce the search area for 3D object by utilizing the frustum projection. However, their detection results highly rely on 2D detection performance.

2D-3D feature fusion methods focus on the early or late 2D and 3D fusion in the process, such as Multi-View 3D networks (MV3D) (Chen et al., 2017), Aggregate View Object Detection (AVOD) network (Ku et al., 2018), 3D semantic instance segmentation (3D-SIS) network (Hou et al., 2019), and Continuous Fusion (ContFuse) network (Liang et al., 2018). MV3D (Chen et al., 2017) proposes the ROI feature fusion using the 2D features extracted from images and 3D features from LiDAR points for the bounding box refinement. AVOD

(Ku et al., 2018) network fuses the 2D and 3D features in both early and late stage, thus, further improves the detection results. ContFuse (Liang et al., 2018) exploits the continuous convolution to concatenate multi-level image and LiDAR features. Discrete state 2D features and continuous geometry are encoded within the continuous fusion layer.

2.3 Evaluation Metrics for Accuracy and Efficiency

To evaluate the proposed methods' accuracy and efficiency for segmentation and detection tasks, several metrics are proposed. These evaluation metrics are also adopted by other published methods for a fair comparison. The detailed description of these metrics is given as follows.

For the segmentation task, the most commonly used evaluation metrics for accuracy are the accuracy, Intersection over Union (IoU) metric, mean IoU (mIoU), and overall accuracy (OA) (Everingham et al., 2015):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

$$IoU_i = \frac{c_{ii}}{c_{ii} + \sum_{j \neq i} c_{ij} + \sum_{k \neq i} c_{ki}} \tag{2.2}$$

$$mIoU = \frac{\sum_{i=1}^{N} IoU_i}{N} \tag{2.3}$$

$$OA = \frac{\sum_{i=1}^{N} c_{ii}}{\sum_{j=1}^{N} \sum_{k=1}^{N} c_{jk}} \tag{2.4}$$

where N is the number of classes, TP, TN, FP, and FN are the number of true positives, true negatives, false positives, and false negatives, respectively. C is an $N \times N$ confusion matrix of the segmentation result, where each entry c_{ii} is the number of points from ground-truth class i predicted as class j. IoU defines the quantify the percent overlap between the target mask and the prediction output. mIoU represents the mean IoU. OA means the proportion of correctly classified points among all the input points.

For 3D object localization and detection task, the most frequently used metrics for accuracy are: Average Precision (AP) and mean average precision (mAP) (Arnold et al., 2019):

$$AP_i = \frac{1}{N}\sum\nolimits_{i \in N} \frac{TP_i}{TP_i + FP_i} \tag{2.1}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{2.2}$$

The AP is used to evaluate the localization and detection performance by calculating the averaged valid bounding box IoU which exceed predefined values. These evaluation metrics are crucial for understanding how applicable the method is in real-world complex scenarios where large quantity number of points must be processed.

Apart from the above evaluation metrics, which are used to measure the 3D information extraction performance in accuracy. The efficiency of deep learning algorithms is commonly evaluated based on training time or inference time within the same experimental settings (Qi et al., 2017). Besides, the memory usage and model size are also referenced as the efficiency evaluation metrics.

2.4 Chapter Summary

In this chapter, the basic knowledge of CNN and point cloud convolution, deep learning based 3D information extraction techniques, and evaluation metrics were systematically reviewed. To understand deep learning segmentation and detection tasks technically, the basic knowledge of CNN and point cloud convolution and the problem definitions of these two tasks were detailly described. A variety of existing deep learning based methods for segmentation, detection, and sensor fusion based object detection were reviewed and analyzed, respectively. It can be concluded through the literature review that the point cloud or graph based deep learning models are more suitable for geometric attributes extraction among 3D space. Besides, the enrollment of local geometric relationship when defining CNNs is a promising direction for discriminate point feature learning. Thus, the segmentation and detection frameworks will be developed based on these two findings in Chapters 3 and 4, respectively. As for sensor

fusion based 3D object detection, the 2D-driven 3D model is considered in Chapter 5 to leverage 2D images for 3D object detection. The corresponding evaluation metrics for quantitative comparison in accuracy and efficiency were followed with the mathematical equation description. These evaluation metrics are partially employed in Chapters 3, 4, and 5 according to the tasks.

Chapter 3

Point Cloud Segmentation

This chapter describes the overall structure of a deep learning based point cloud segmentation algorithm. In Section 3.1, the background of the deep learning based semantic segmentation, the preliminary knowledge of geometric convolution, and the implementation details of the proposed algorithm are described. In Section 3.2, experimental settings, including the selected datasets, evaluation metrics, segmentation results, ablation studies, and the timing and memory usage are provided in detail. Section 3.3 discusses the quantitative and qualitative results of the proposed framework. Section 3.4 provides a summary of this chapter.

This chapter is mainly a paper published in a journal and only minor format changes have been made in order to make them to fit into the format of the entire thesis.

3.1 Algorithm Description

Semantic segmentation is urgently desired for a comprehensive scene understanding in real-time perception and urban modeling (Tchapmi et al., 2017). Similar to per-pixel image labeling, 3D semantic segmentation seeks to attribute a semantic classification label to each 3D point. Given the features are hierarchically learned in an end-to-end trainable framework (Qi et al., 2017), deep convolutional neural network (CNN) models have achieved remarkable success in 2D semantic segmentation tasks. However, compared with 2D regular imagery data, 3D point clouds are uneven, unstructured, noisy, and irregular data, which cannot exploit the classical 2D CNNs directly.

Recently, several methods have been proposed to define convolution filters in non-Euclidean domain, which can directly process irregular data such as point clouds. These

approaches, prompting the emerging field of geometric deep learning (Bronstein et al., 2017), can be roughly classified into two types: spectral-based and spatial-based methods. The spectral-based methods define the convolution operations by exploiting the spectral eigen-decomposition of the graph Laplacian (Yi et al., 2017). The signal frequencies of the graph constructed from point clouds are commonly represented by the eigenvalues of the graph Laplacian. They are filtered in the spectral domain, similar to the Fourier domain filtering of conventional signals (Fey et al., 2018). But these spectral-based geometric CNNs have the following two problems: (1) the learned spectral filter's coefficients are not suitable for another domain with a different basis (Bronstein et al., 2017); and (2) the spectral filtering is calculated based on the whole input data, which requires high computation capability.

Thus, Masci et al. (2015) proposed the first spatial-based CNN on non-Euclidean data, applying filters to local neighbours represented in geodesic polar coordinates. Qi et al. (2017) constructed spatial-based CNNs by defining convolution kernels in local neighbours with respect to local Euclidean positional relationships between points. Monti et al. (2017) defined the convolution kernels based on the degrees of the nodes. Then these learned features are aggregated (e.g., sum or max) to generate new point or vertex feature vectors. Compared with spectral-based CNNs, spatial-based CNNs are not basis-dependent and, thus, can be transformed into different domains (Bronstein et al., 2017). In addition, spatial filtering that is conducted in the local region has a lower computation cost. However, the aforementioned spatial-based CNNs suffer the following two limitations.

- The high-level geometric correlations between the input and its neighbouring coordinates or features are not fully exploited in defining convolution kernels. These correlations can enhance the kernel's shape description capability.
- The traditional aggregation functions, e.g., max or mean, discard or neglect the structural connection among local neighbours because different neighbours contribute differently.

To address the above two challenges, we propose an alternative geometric graph convolution, termed TGConv, which is designed to explore high-level geometric correlations among local neighbours extracted from point clouds for semantic segmentation. These filters

are defined as products of local neighbour point features with geometric features extracted from local coordinates expressed by a family of Gaussian weighted Taylor kernel functions. Although local coordinates can express the low-level geometric characteristic for local neighbours, we use our defined functions to map the position information to high-level geometric attributes. Then a parametrized pooling operation based on distance metric is proposed for effective feature aggregation. Such aggregation is composed of the max and a learnable distanced-based weight function, which can harness the most representative features and adaptively exploit related neighbour features.

Based on the proposed TGConv, we construct an end-to-end geometric graph convolution architecture on the graph representation of a point cloud, called Taylor Gaussian mixture model (GMM) network (TGNet). To improve the scale invariance of our network, TGNet employs a multiscale hierarchical architecture by operating TGConv on neighbourhoods at multiple scales, which allows it to extract coarse-to-fine semantic deep features. Besides, a conditional random field (CRF) layer is combined within the output layer to further improve the segmentation result.

3.1.1 Preliminary Knowledge

The convolution in a Euclidean domain can be defined as extracting a template patch at each point of the domain and learning the correlation of the patch with the function at that point. Thus, for 2D imagery convolution in regular Euclidean domain, per-pixel patch extraction at each position is always the same. However, due to the unstructured and irregular data structure of point clouds and the different input shapes, it is difficult to define an effective convolution operation in non-Euclidean domains. There are two requirements in the construction of non-Euclidean CNNs which are as follows.

- The local patch extraction should be shift-invariant; however, it is actually position-dependent.
- The patch has to be represented in a local intrinsic coordinate system because of the difficulty in global parametrization in non-Euclidean domains.

To achieve these, Monti et al. (2017) and Kipf and Welling (Kipf and Welling, 2016) constructed patch operators $D(\cdot)$ by defining a family of learnable weight functions

$w_1(u), \cdots, w_J(u)$ of a local patch (e.g., a local graph) represented by pseudo-coordinates u. Given vertex x and its neighbour [denoted as $x' \in \mathcal{N}(x)$] features f , the patch operator can be formulated as the weighted summation of f:

$$D(x)f = \sum_{x' \in \mathcal{N}(x)} f(x') w_j(u(x, x')) \, , j = 1, \cdots, J \tag{3.1}$$

Based on the above fact, a spatial geometric convolution on non-Euclidean domains is defined as:

$$(f * g)(x) = \sum_{j=1}^{J} g_\theta (D_j(x) f) \tag{3.2}$$

where $*$ represents the convolution operation, g_θ denotes the learnable coefficients applied on the patch extracted at each point.

This kind of geometric convolution kernels has been applied in several non-Euclidean CNNs such as Graph Convolutional Networks (GCN) (Kipf and Welling, 2016) and mixture model network (MoNet) (Monti et al., 2017) by defining different weight functions. However, these methods just use the local intrinsic coordinate information, the high-level geometric feature is not fully exploited, which is crucial for robust semantic segmentation. Besides, the traditional aggregation method such as max, sum, or mean pooling operation is not adaptable for various inputs. To solve the above two challenges, we define our TGConv as a product of local neighbour point features with geometric features extracted from local coordinates expressed by a family of Taylor kernel functions. In addition, we proposed a learnable pooling function to aggregate features to improve the performance of discriminative feature learning.

3.1.2 TGConv

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathrm{U})$ constructed from a given 3D point cloud $\mathrm{P} = \{p_1, p_2, \cdots, p_n\} \in R^3$ according to their spatial neighbours, where $\mathcal{V} = \{1, 2, \ldots, \mathrm{n}\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represent the set of vertices and edges, respectively, and U contains 3D pseudo-coordinates $\mathrm{u(x, y)} \in R^3$ for each directed edge $(\mathrm{x, y}) \subseteq \mathcal{E}$. Denote each point $y \in \mathcal{N}(x)$ as the neighbour set of vertex x, $\mathrm{u(x, y)}$ is a 3D vector of pseudo-coordinates for each y. Let $\mathrm{h} =$

$\{h_1, h_2, \cdots, h_n\}$ be a set of input vertex features, each feature $h_1 \in R^F$ is associated with a corresponding graph vertex i $\subseteq \mathcal{V}$, where F is the feature dimension of each vertex.

To leverage spatially local correlation, we mimic Eq.**(3.1)** and Eq.**(3.2)** to conduct local operations on the local graph, by parametrizing a family of convolutional filters. These filters are defined as products of local neighbour point features with geometric features extracted from local coordinates expressed by a family of Gaussian weighted Taylor kernel functions (see **Figure 3.1**). Then they are aggregated via a parametric pooling operation to new point set features $h' = \{h'_1, h'_2, \cdots, h'_n\}$ with $h'_i \in R^K$.

In Eq.**(3.1)**, the patch operator is defined directly on the pseudo-coordinates u(x, y). Although geometric information can be extracted, however, high-level geometric spatial features are not exploited. Thus, we map the pseudo-coordinates to a high-level geometric feature using a function T (u): $R^3 \rightarrow$ R, which

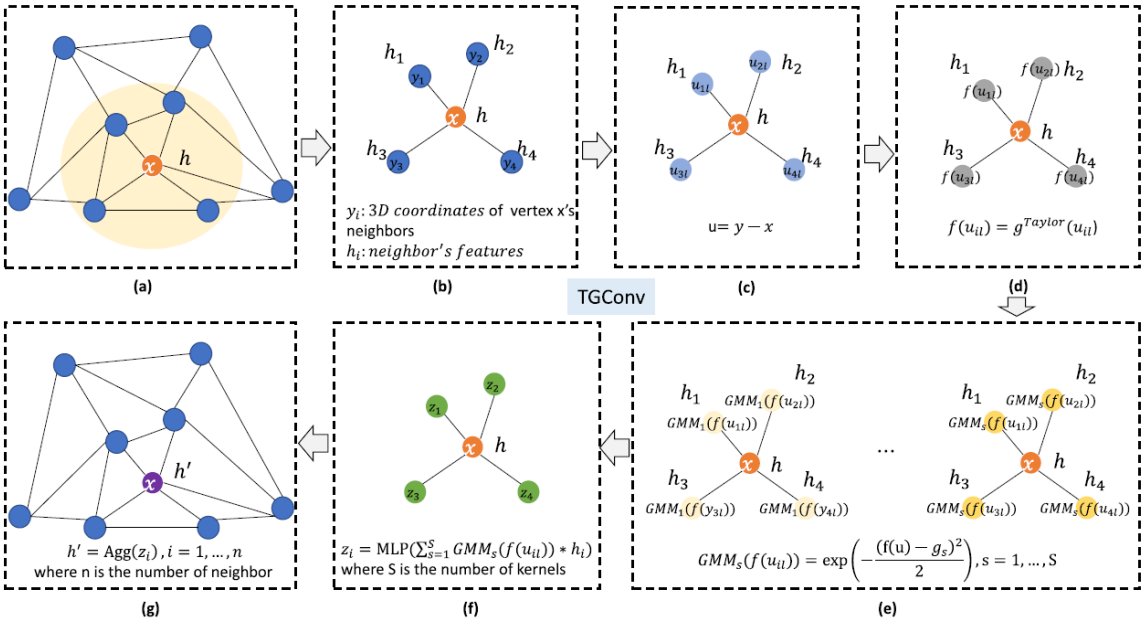

Figure 3.1: TGConv on graph representation of point clouds.

can improve the geometric expression of the patch operator. Besides, the summation is not suitable for aggregating the effective and robust features. To solve this, we define a learnable

aggregation function to adaptively pool local features. As a result, we define our convolution operation as:

$$(f * g)(x) = \mathrm{Agg}(g_\theta(\sum_{s=1}^{S} D_s(x)h_y), \qquad y \in \mathcal{N}(x) \tag{3.3}$$

where $\mathrm{Agg}(\cdot)$ represents the aggregation function, $g_\theta(\cdot)$ is the learnable feature mapping function: $R^F \to R^K$. The weight function Ds $(\cdot)$ is defined as

$$D_s(x) = w_s(T(u)), \qquad s = 1, \cdots S \tag{3.4}$$

with u representing the pseudo-coordinates,$\mathrm{w(T)} = w_1(T), \cdots, w_S(T)$ being weight functions parametrized by learnable parameters, and S being the number of kernels.

The critical construction of our proposed kernels is the choices of the pseudo-coordinates u, geometric pseudo-coordinates mapping function $T(u)$, weight functions w(T), feature mapping function $g_\theta(\cdot)$, and aggregation function $\mathrm{Agg}(\cdot)$.

Pseudo-coordinates: Pseudo-coordinates, such as polar, spherical, or Cartesian coordinates, encode local positional relationships between points (Fey et al., 2018) and can be used to describe local geometric features. **Table 3.1** lists the selection of pseudo-coordinates u and weight function w(u) of some geometric deep learning methods (Kipf and Welling, 2016; Monti et al., 2017; Qi et al., 2017; Wang et al., 2019). For example, MoNet (Monti et al., 2017) and GCN (Kipf and Welling, 2016) define their kernels on the pseudo-coordinates based on the degree of graph vertices. PointNet++ (Qi et al., 2017) and DGCNN (Wang et al., 2019) select the local Euclidean coordinates as their pseudo-coordinates. In order to reduce the computation cost and exploit the original geometric feature from 3D coordinates, local Euclidean coordinates are selected as our pseudo-coordinates. For each vertex x and vertex $y \in \mathcal{N}(x)$ in the neighbourhood of x, we consider local pseudo-coordinate $\mathrm{u(x, y)}$ as:

$$\mathrm{u(x, y)} = \mathrm{y} - \mathrm{x} = (u_1, u_2, u_3)^T \tag{3.5}$$

where each vertex x is represented by 3D Cartesian coordinates, and $(u_1, u_2, u_3)^T$ represent the corresponding pseudo-coordinate along each axis of each neighbourhood point y to point x.

Table 3.1: Choice of pseudo-coordinates and weight functions of several geometric CNN models

Method	Aggregation	Pseudo-coordinates	Pseudo-coordinates u(x, y)	Weight Function $w_s(u)$, $s = 1, \cdots S$
PointNet++	max	Local Euclidean	$\mathrm{u(y)} - \mathrm{u(x)}$	-
MoNet	$\sum$	Vertex degree	$(\frac{1}{\sqrt{\deg(x)}}, \frac{1}{\sqrt{\deg(y)}})$	$\exp\left(-\frac{1}{2}\left(u-\mu_j\right)^T \sum_j^{-1}\left(u-\mu_j\right)\right)$
GCN	$\sum$	Vertex degree	$(\deg(\mathrm{x}), \deg(\mathrm{y}))$	$\left(1-\left\lvert 1-\frac{1}{\sqrt{u_1}}\right\rvert\right)\left(1-\left\lvert 1-\frac{1}{\sqrt{u_2}}\right\rvert\right)$
DGCNN	max	Local Euclidean	$\mathrm{u(y)} - \mathrm{u(x)}$	-

Geometric Pseudo-coordinates Mapping Function: The pseudo-coordinates leverage only the low-level spatial information, and the high-level structural and geometric information among pseudo-coordinates is not exploited. Based on that, we design our filters considering the high-level structural information of pseudo-coordinates to increase the CNN kernels' shape description ability. To ensure that the filters are powerful enough to extract intricate local geometric features, a mapping function T(u) is used to leverage the intrinsic information among $(u_1, u_2, u_3)^T$ into a high-level representation $\mathrm{g}(\boldsymbol{u_1}, \boldsymbol{u_2}, \boldsymbol{u_3})$. There are two important considerations for choosing such a mapping function T(u): 1) the optimization is convenient to conduct and 2) it can interpolate arbitrary values in local graph. Inspired by SpiderCNN (Xu et al., 2018), the order-3 Taylor expansions of 3D coordinates are used to map the local pseudo-coordinates u into the high-level geometric attribute.

$$\mathrm{T(u)} = g_{\sigma^T}^{Taylor}(\boldsymbol{u_1}, \boldsymbol{u_2}, \boldsymbol{u_3}) = \sigma_0^T + \sum_{i=1}^{3} \sigma_i^T \boldsymbol{u_i} + \sum_{i=1}^{3} \sum_{j=1, i\le j}^{3} \sigma_{ij}^T \boldsymbol{u_i u_j} + \sum_{i=1}^{3} \sum_{j=1}^{3} \sum_{\substack{k=1 \\ i\le j\le k}}^{3} \sigma_{ijk}^T \boldsymbol{u_i u_j u_k} \tag{3.6}$$

where $\sigma_0^T, \sigma_i^T, \sigma_{ij}^T$ and σ_{ijk}^T are the 1×1 learnable parameters. By varying these parameters, $\mathrm{g}(\boldsymbol{u_1}, \boldsymbol{u_2}, \boldsymbol{u_3})$, we can approximate arbitrary values.

Weight function: In MoNet (Monti et al., 2017), a Gaussian mixture model (GMM) is used as the weight functions w(u) with learnable parameters as:

$$w_s(u) = \exp\left(-\frac{1}{2}\left(u - \mu_s\right)^T \sum\nolimits_s^{-1} (u - \mu_s)\right), \qquad s = 1, \cdots S \tag{3.7}$$

where $\sum_s$ and μ_s are learnable $\mathrm{d} \times \mathrm{d}$ and $\mathrm{d} \times 1$ covariance matrix and mean vector of a Gaussian kernel, respectively. Their experimental results have demonstrated that they can distinguished performance. Thus, we also adopt the GMM as our weight functions. Because we have mapped our pseudo-coordinates into a more powerful geometric feature with one dimension, our weight function is defined as:

$$w_s(u) = \exp\left(-\frac{1}{2}(g - g_s)^2\right), \qquad s = 1, \cdots S \tag{3.8}$$

where g_s are learnable 1×1 mean vector of a Gaussian kernel. The kernel number S is experimentally set to 10 in our method.

Based on the above function, we can get our intermediate feature h'_m within the input feature h:

$$h'_m = \sum_{s=1}^{S} \exp\left(-\frac{1}{2}\left(g_{\sigma^T}^{Taylor}(\boldsymbol{u}) - g_s\right)^2\right) h \tag{3.9}$$

Compared with traditional CNNs, these convolution kernels can better exploit the learnable 3D geometric features and are easy to optimize.

Feature Mapping Function: The feature mapping function $g_\theta(\cdot)$ is applied on each vertex to map the intermediate feature h'_m from R^F to R^K. In our article, $g_\theta(\cdot)$ is a multilayer perception (MLP). Because, theoretically, an MLP with one hidden layer can approximate an arbitrary continuous function (Hornik, 1991). Besides, MLP retains the crucial characteristic of standard convolution in grid domain: weight sharing. Thus, the input intermediate feature h'_m is mapped as:

$$h'_\theta = MLP(h'_m) \tag{3.10}$$

Aggregation Function: Aggregation operation aims to output the aggregated features on the vertices of a coarsened graph. Traditionally, the most commonly used pooling function is max function (Qi et al., 2017), which corresponds to the max pooling. The main reason is that the most discriminate feature can better represent the local pattern. However, max pooling operation discards some other fine-grained features which results a coarse prediction for semantic segmentation. To better leverage the most discriminate features and local contextual features, in this article, we use the max and a learnable weighted average function for graph pooling and concatenate these two pooling results as the output aggregated features. Thus, the output feature set for vertex x is calculated as follows:

$$h'_x = \max\{h'_{\theta y}\} + \frac{\sum_{j=1}^{k} \theta_j \omega_j h'_{\theta y}}{\sum_{j=1}^{k} \omega_j}, \qquad y \in \mathcal{N}(x) \tag{3.11}$$

where

$$\omega_j = \frac{1}{d(p_y, p_x)^P} \tag{3.12}$$

and p_y, and p_x represent the coordinates of neighbour point y and the vertex x. k represents the number of neighbours. θ_j is a learnable 1×1 vector, which is used to learn most relevant neighbour features and reweight the nonrelevant neighbour features with low values even if they are close to the vertex. The distance metric p is set to 2 in our experiment. This aggregation method improves the discriminative capability of the network by considering nearby neighbours' features to influence prediction.

3.1.3 Taylor GMM Convolutional Network

Our TGNet builds a graph pyramid of point clouds by hierarchically grouping the points and progressively abstracting larger and larger local regions along the hierarchy, as shown in **Figure 3.2**. At each scale of the graph pyramid, TGConv is applied for local feature learning. After that, the learned features are interpolated back to the finest scale layer by layer. Similar to PointNet++ (Qi et al., 2017), features at the same scale are skip-connected. Besides, due to the limitation of computation, TGConv can only be applied to the sampled input features which cannot provide fine-grained per-point information for semantic segmentation. Thus, a shared MLP is applied to the raw input to extract per-point features. These learned features are combined with the interpolated features learned from the finest layer to predict the per-point semantic label likelihood. Finally, considering the loss of feature fidelity caused by the multiple graph pooling and feature interpolation layers, an additional CRF layer is applied at the finest scale for feature refinement.

Graph Sampling and Grouping Module: In order to increase the receptive field of TGConv, the raw input point clouds are hierarchically subsampled into different scales. We use the farthest point sampling (FPS) algorithm (Qi et al., 2017) to subsample the point set with a family of ratios. Given the input point set P, FPS iteratively selects a subset of points which is the most distant point from this set compared with the remaining points. This method is data-dependent and adaptive to various point clouds with uneven density. Thus, within the input point set P, the subsampled point clouds are denoted as $P_1, \cdots, P_L$ where L represents the number of scales. For each $P_l (l = 0, \cdots, L)$, a corresponding graph $\mathcal{G}_l = (\mathcal{V}_l, \mathcal{E}_l)$ can be constructed as described above.

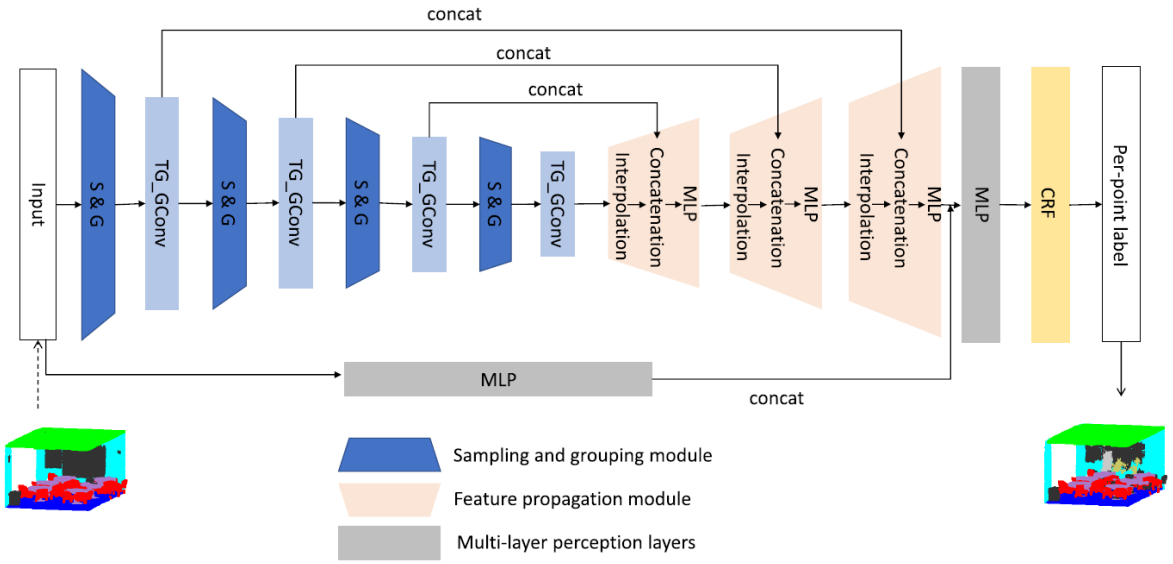

Figure 3.2: Framework of our TGNet.

Because our TGConv is operated in the local region for each vertex at multiple scales. Thus, how to find spatially important neighbours is of great significance. There are two ways to search the nearest neighbours: Spherical neighbourhood (Thomas et al., 2018) and K-nearest neighbour (KNN) (Engelmann et al., 2018). The first one selects *k* neighbours randomly within radius, while KNN chooses the point with k smallest distance neighbours among all the input points. Thus, spherical neighbourhood is adaptive to density variation. Because the point clouds are commonly distributed unevenly, the spherical neighbourhood is selected to enhance the framework's invariance to density change. We determine the radius of the local spherical neighbour experimentally. Our method processes both indoor and outdoor scenes. However, points in indoor scenes scanned by RGB-D camera have uniform point distribution, while point clouds of outdoor scenes acquired by an MLS system have irregular and sparse point density. Fixed radius is simple and cost-effective, but not adaptive-efficient. An adaptive radius is effective but not cost-efficient. Based on the above characteristics and computation cost, we set each sampling radius to a fixed value determined experimentally.

Feature Propagation Module: Although the hierarchical sampling can improve the receptive field of TGConv, the fine-grained information is lost. Besides, semantic labeling

needs the feature for each point. Thus, to obtain a distinctive result, the interpolation of learned point features between the coarsest to raw scale must be conducted gradually. Let h_l be the learned feature set at the lth scale of the graph pyramid, P_l and P_{l-1} are the spatial coordinates set of the lth and $l-1$th scales, respectively. To obtain features at the $l-1$th scale, we use the inverse distance weighted average based on KNNs (denoted as $p_i, i = 1, \cdots, k$) for each point p of P_{l-1} in P_l (see (13), k = 3, q = 2) to calculate the weighted sum of their features:

$$h_{l-1} = \frac{\sum_{i=1}^{k} \theta_j h_j}{\sum_{i=1}^{k} \omega_j}, \qquad where\ \omega_i(x) = \frac{1}{d(p, p_i)^q} \tag{3.13}$$

These interpolated features on P_{l-1} are then concatenated with skip linked point features from the corresponding TGConv layer. Then a shared MLP is applied to these concatenated features using 1×1 CNNs to update each point's features.

CRF Layer: CRF (Zheng et al., 2015) is commonly applied to postprocess the CNN's prediction results in semantic segmentation challenges. Because convolutional filters with large receptive fields produce coarse semantic results for each point. CRF inference formulates the label assignment task as the probabilistic inference problem, which encourages spatially close and appearance-similar points to share consistent labels. Thus, CRF can help to refine our weak and coarse point-level labeling results. However, CRF is commonly applied in the post-process step, which cannot fully exploit the advantage of the CRF, because it is not integrated with neural networks. To harness it in deep learning frameworks, in (Krahenbuhl and Koltun, 2011), an approximate inference method is proposed. It assumes independence between semantic label distributions $\mathrm{Q(X)} = \Pi_i Q_i(x_i)$, an derives the update equation:

$$\begin{aligned} Q_i^+(x_i = l) = \frac{1}{Z_i} exp \Bigg\{ & -\psi_u(x_i) \\ & - \sum_{l' \in \mathcal{L}} \mu(l, l') \sum_{m=1}^{K} \omega^{(m)} \times \sum_{j \neq i}^{K} k^{(m)}(f_i, f_j) Q_j(l') \Bigg\} \end{aligned} \tag{3.14}$$

Based on that, Zheng et al. (Zheng et al., 2015) formulated CRF inference and learning as a RNN, termed CRFasRNN. We integrate this CRF layer following our TGNet framework for

joint training and inference. Thus, the coarse semantic labeling results can be further improved in a learnable scheme.

3.2 Experiments

To verify the effectiveness of our proposed algorithm, qualitative and quantitative evaluations were conducted on indoor and outdoor point cloud data sets, including ScanNet data set (Dai et al., 2017), Stanford Large Scale 3D Indoor Spaces (S3DIS) data set (Armeni and Zamir, 2016) and Paris-Lille-3D data set (Roynard et al., 2018). Before we conduct experiments on the above three data sets, some ablation studies of TGNet are first analyzed to demonstrate the effectiveness of our method.

3.2.1 Data Sets

ScanNet (Dai et al., 2017): The ScanNet data set contains 1513 scans by using RGB-D video streaming in indoor environments, such as offices, apartments, conference rooms, etc. These scans are split into 1201/312 scenes for training/testing in semantic voxel labeling. This data set was manually interpreted and labeled into 20 classes, such as the floor, desk, curtains, and bathtubs.

S3DIS Data Set (Armeni and Zamir, 2016): The S3DIS data set was generated from three different buildings which contain five large-scale indoor areas, covering a total of 6020 m^2. These scenes have different architectural styles and appearances, including offices, conference rooms, open spaces, etc. The whole data set was manually labeled with 12 semantic elements according to their attributes, e.g., structural elements, common indoor items, and furniture. Each point is represented by a nine-dimension vector of XYZ, RGB, and normalized location.

Paris-Lille-3D (Roynard et al., 2018): Paris-Lille-3D data set contains 140 million points and covers 55,000 m^2 area in outdoor environments. This data set was acquired by a MLS system in two cities: Paris and Lille. Thus, the points in this data set are sparse and relatively low measurement resolution compared with the above two indoor data sets. The whole data set was fully annotated into 50 classes unequally distributed in three scenes: Lille1, Lille2, and Paris. For simplicity, these 50 classes were combined into ten coarse classes for challenging.

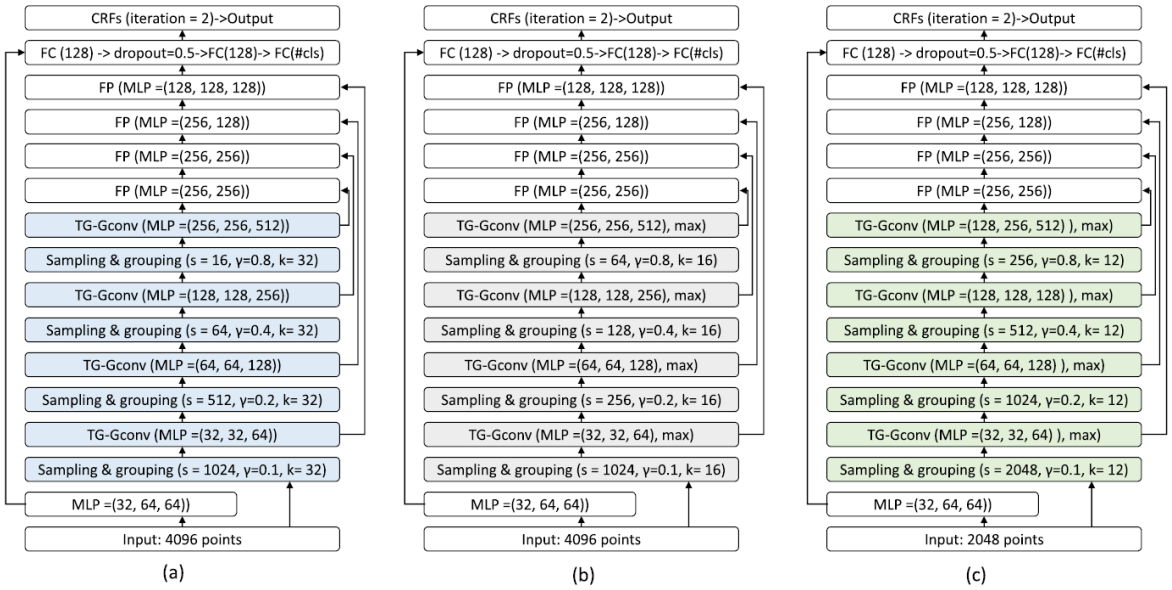

Figure 3.3: TGNet model zoo for ScanNet, S3DIS, and Paris-Lille-3D data sets.

3.2.2 Evaluation Metrics

On ScanNet data set, we adopted accuracy and unweighted average accuracy (Qi et al., 2017) as our evaluation metric, which is different from the overall accuracy (OA) used in PointWeb (Zhao et al., 2019). OA means the proportion of correctly classified points among all the input points (Story and Congalton, 1986), while unweighted average accuracy represents the unweighted average of each accuracy per class. Because there exit biases between different semantic classes in real scene. Points with large proportion have high probability to be learned and predicted correctly. Objects with low proportion are generally hard to be labeled accurately. To demonstrate the effectivity of our TGNet that can learn and distinguish small or uncommon objects, we selected unweighted average accuracy as our evaluation metric.

On S3DIS and Paris-Lille-3D data sets, three metrics, including per-class intersection over union (IoU) (Wang et al., 2019), mean IoU (mIoU) of each class (Wang et al., 2019), and OA were employed to quantitatively evaluate the performance of our method. IoU evaluates per-class segmentation result, while mIoU can reflect the average segmentation result considering all semantic classes.

3.2.3 Ablation Studies and Analysis

Table 3.2: Ablation studies on ScanNet test set

Ablation studies	**Avg (%)**
Aggregation methods	
Base model	57.8
TGNet (with max pooling)	61.6
TGNet (with max + parametric weighted average pooling)	62.2
CRFasRNN	
TGNet (without CRF)	59.7
CRFasRNN (1 iteration)	61.4
CRFasRNN (2 iteration)	62.2
CRFasRNN (5 iteration)	61.0
Number of nearest neighbours	
K=8	56.8
K=16	60.0
K=24	60.5
K=32	62.2

In order to verify the effectiveness of our proposed aggregation method, CRFasRNN, and determine the number of nearest neighbours, we conduct several ablation studies on ScanNet test data set (Dai et al., 2017) and show their results in Table 3.2.

1) Ablation Test of Aggregation Methods: Our base model is PointNet++ (Qi et al., 2017), which achieves 57.8 % unweighted average accuracy. To demonstrate the effectiveness of our proposed aggregation method in TGNet, we test two different aggregation methods: max pooling and max and parametric weighted average pooling. Specifically, we only replace the

max and parametric weighted average pooling in TGConv with the max operator while keeping the rest unchanged in our TGNet. We can see that the average accuracy of our TGNet is 0.8% higher than the max pooling aggregation method, which shows that our max and parametric weighted average pooling has more advantages in discriminative feature learning than the max operator. Because max operator only considers the most representative features, however, the remaining features are actually contributed differently to feature learning. Our proposed aggregation method not only consider the most representative features but also learn the remaining features based on their spatial location and adapt learned features via parameter optimization. Compared with the base model, our TGNet improves 4.4% average accuracy in semantic voxel labeling task.

2) CRFasRNN: CRF is commonly used to improve the segmentation results by adding smoothness constraints between points which have similar features. In GACNet (Wang et al., 2019), graph attention convolution (GAC) is applied to process the finest scale points in the last layer. However, due to the computation limitation, we cannot apply TGConv to the same layer. Thus, in the last layer, CRF in our framework plays a similar role as GAC to consider weights in more relevant parts. To experimentally verify its effectiveness in our model, we add CRFasRNN layer in the last layer of our TGNet using different iterations. Specifically, we use the Gaussian kernels from (Zheng et al., 2015) for the pairwise potentials of CRF. The testing results on the ScanNet test data set are also provided in **Table 3.2** for comparing convenience. We can see that, within the integration of CRFasRNN layer, the average accuracy of semantic segmentation result is improved about 1.8% compared with TGNet without CRFasRNN layer. With two iterations, the CRFasRNN has basically converged, and more iterations do not result in considerably increased accuracy. Thus, in our TGNet, the iteration number is set to 2.

3) Effect of the Number of Nearest Neighbours: We also study the number of nearest neighbours k chosen in TGNet, where the results are provided in **Table 3.2**. The number of nearest-neighbours k is analogous to the size of the receptive field in the common convolution. 32 is the optimal choice, achieving the 62.2% unweighted average accuracy, among 8, 16, 24, and 32-nearest neighbours. The higher number of nearest neighbours is not tested due to the limitation of our computation capability.

3.2.4 Segmentation Results

Semantic Voxel Labeling on ScanNet: There are 1201/312 scenes in ScanNet for training/testing our TGNet. The framework and implementation details of this network are depicted in Figure 3.3. The input to the network is 4096 points with XYZ information. The sampling point in each layer is: 1024, 512, 64, 16, and the number of nearest neighbours is experimentally set to 32. Due to the limitation of computation capability, the TGConv is only applied in these subsampled points. We use max and parametric weighted pooling in our TGConv filters. As for the MLP layers, we use 1×1 convolution kernels to process the extracted features. The training epoch is set to 200.

Table 3.3 lists quantitative results of our semantic segmentation on a voxel-basis for 20 classes. Our method achieves the highest unweighted accuracy of 62.2%. Most objects can be correctly labeled, except picture, cabinet, door, window, counter, and desk objects. These six objects only occupy a limited ratio of the whole scene, or share a similar shape with other objects, thus their poor segmentation results reduced the unweighted average accuracy. Compared with several existing methods, e.g., ScanNet (Dai et al., 2017), Scan-Complete (Dai et al., 2018), 3DMV (Dai and Niesner, 2018), Recurrent Slice Networks (Huang et al., 2018), PointNet (Qi et al., 2017), FCPN (Rethage et al., 2018), PointNet++ (Qi et al., 2017), and Matter-port3D (Chang et al., 2017), our approach learns geometry features hierarchically. This is crucial for understanding scenes at different scales and labeling objects with different sizes. Although PointNet++ (Qi et al., 2017) learns hierarchical and geometry features at different scales, the geometric coordinates are not applied in defining their convolutions. Therefore, its performance on small or uncommon objects is suboptimal. We can note that the improvement of our TGNet mainly comes from uncommon or shape-similar objects, e.g., sofa, curtain, and window. Besides, our framework is based on PointNet++ (Qi et al., 2017), we have tested their published code in our own computer on this data set and achieved 57.8% unweighted accuracy, shown in Table 3.2. This can further demonstrate the effectiveness of our method.

Table 3.3: Semantic voxel label prediction accuracy (%) on ScanNet test scenes

Method	Of the scenes	ScanNet	Scan-Complete	3DMV	Recurrent Slice Networks	PointNet	FCPN	PointNet++	Matter-port3D	Ours
Wall	38.8	70.1	87.2	60.4	79.2	69.4	87.7	89.5	78.8	79.7
Floor	35.7	90.3	96.9	95.0	94.1	88.6	96.3	97.8	92.6	96.6
Cab	2.4	49.8	44.5	54.4	31.3	5.0	52.1	39.8	91.1	46.6
Bed	2.0	62.4	65.7	69.5	56.0	18.0	65.9	80.7	60.6	81.1
Chair	3.8	69.3	75.1	79.5	65.0	35.9	81.6	86.0	20.7	82.2
Sofa	2.5	75.7	72.1	70.6	55.4	32.8	76.0	68.3	28.4	85.3
Table	3.3	68.4	63.8	71.3	51.0	32.8	67.6	59.6	14.4	64.8
Door	2.2	48.9	13.6	65.9	3.0	0.0	27.5	16.6	14.7	29.0
Wind	0.4	20.1	16.9	20.7	8.8	0.0	12.5	23.7	0.0	36.9
Bkshf	1.6	64.6	70.5	71.4	53.0	3.2	81.0	84.3	1.0	83.5
Pic	0.2	3.4	10.4	4.2	1.0	0.0	1.8	0.0	7.5	0.0
Cntr	0.6	32.1	31.4	20.0	22.7	5.1	31.6	37.6	23.8	33.0
Desk	1.7	36.8	40.9	38.5	34.5	2.6	58.5	66.7	54.0	42.2
Curt	0.7	7.0	49.8	15.2	6.8	0.0	6.1	48.7	85.4	69.3
Fridg	0.3	66.4	38.7	59.9	37.9	0.0	54.7	54.7	6.8	60.6
Show	0.04	46.8	46.8	57.3	29.9	0.0	48.0	85.0	20.2	84.9
Toil	0.2	69.9	72.2	78.7	54.2	0.0	86.7	84.8	5.1	89.4
Sink	0.2	39.4	47.4	48.8	34.8	0.0	53.5	62.8	27.5	70.6
Bath	0.2	74.3	85.1	87.0	49.4	0.2	79.1	86.1	18.3	89.4
other	2.9	19.5	26.9	20.6	19.0	0.1	30.2	30.7	16.6	15.7
avg	-	50.8	52.8	54.4	48.4	19.9	54.2	60.2	33.4	**62.2**

Semantic Segmentation on S3DIS: Although there are six labeled indoor areas in S3DIS data set, for a principled evaluation, the Area 5 is selected as our testing set and the rest is used to train our TGNet (Qi et al., 2017; Tchapmi et al., 2017; Landrieu and Simonovsky, 2018; Ye et al., 2018; Wang et al., 2019; Wang et al., 2019). Notably, Area 5 is not in the same building as other areas, and there exist some differences between the objects in Area 5 and other areas. This across-building experimental setup is better for measuring the model's generalizability, while also brings challenges to the segmentation task.

The input to the network is 4096 points with nine-dimension features in training and testing our model. The sampling point in each layer is 1024, 256, 128, 64, and the number of nearest neighbours is experimentally set to 16. The TGConv is only applied to the above layers. Because the S3DIS has larger data than ScanNet and we have limited computation capability, we replace the aggregation function as max pooling operation. The other experimental setting is the same as the ScanNet framework. The training epoch is set to 100.

The quantitative evaluations of the experimental results are provided in Table 3.4. We can see that our TGNet achieves the best OA than other competitive methods, e.g., PointNet (Qi et al., 2017), SegCloud (Tchapmi et al., 2017), 3P-RNN (Ye et al., 2018), SPG (Landrieu and Simonovsky, 2018), DGCNN (Wang et al., 2019), and GACNet (Wang et al., 2019). As the convolution weights of TGConv are assigned according to not only the spatial positions but also the geometric attributes of the neighbouring points, the proposed TGNet is able to capture the discriminative feature of point clouds even though the spatial geometry is lost or weak. However, our mIoU is lower than the result of GACNet (Wang et al., 2019). We guess the main reason is that they applied GAC in the first and last layers which have 4096 points and thus acquired more accurate per-point features for segmentation.

In Figure 3.4, semantic segmentation results of S3DIS within 6 representative scenes are presented. Compared to groundtruth, most areas can be accurately predicted. But in the connected area of several different objects, the predicted boundary is unclear and blurred. This is mainly due to the limited receptive field of TGConv constrains its geometric feature learning ability to differentiate connected objects.

Table 3.4: OA (%) and mIoU (%) on S3DIS data set.

	PointNet	SegCloud	3P-RNN	SPG	DGCNN	GACNet	Ours
OA	-	-	-	86.4	59.8	87.8	**88.5**
mIoU	41.1	48.9	53.4	58.0	51.5	**62.9**	57.8
Ceiling	88.8	90.1	95.2	89.4	93.0	92.3	93.3
Floor	97.3	96.1	98.6	96.9	97.4	98.3	97.6
Wall	69.8	69.9	77.4	78.1	77.7	81.9	78.0
Beam	0.1	0.0	0.8	0.0	0.0	0.0	0.0
Column	3.9	18.4	9.8	42.8	12.2	20.4	9.3
Window	46.3	38.5	52.7	48.9	47.8	59.1	57.0
Door	10.8	23.1	27.9	61.6	39.8	40.9	39.4
Chair	52.6	75.9	76.8	84.7	67.4	78.5	83.4
Table	58.9	70.4	78.3	75.4	72.4	85.8	76.4
Bookcase	40.3	58.4	58.6	69.8	23.2	61.7	60.6
Sofa	5.9	40.9	27.4	52.6	52.3	70.8	41.8
Board	26.4	13.0	39.1	2.1	39.8	74.7	58.7
Clutter	33.4	41.6	51.0	52.2	46.6	52.8	55.3

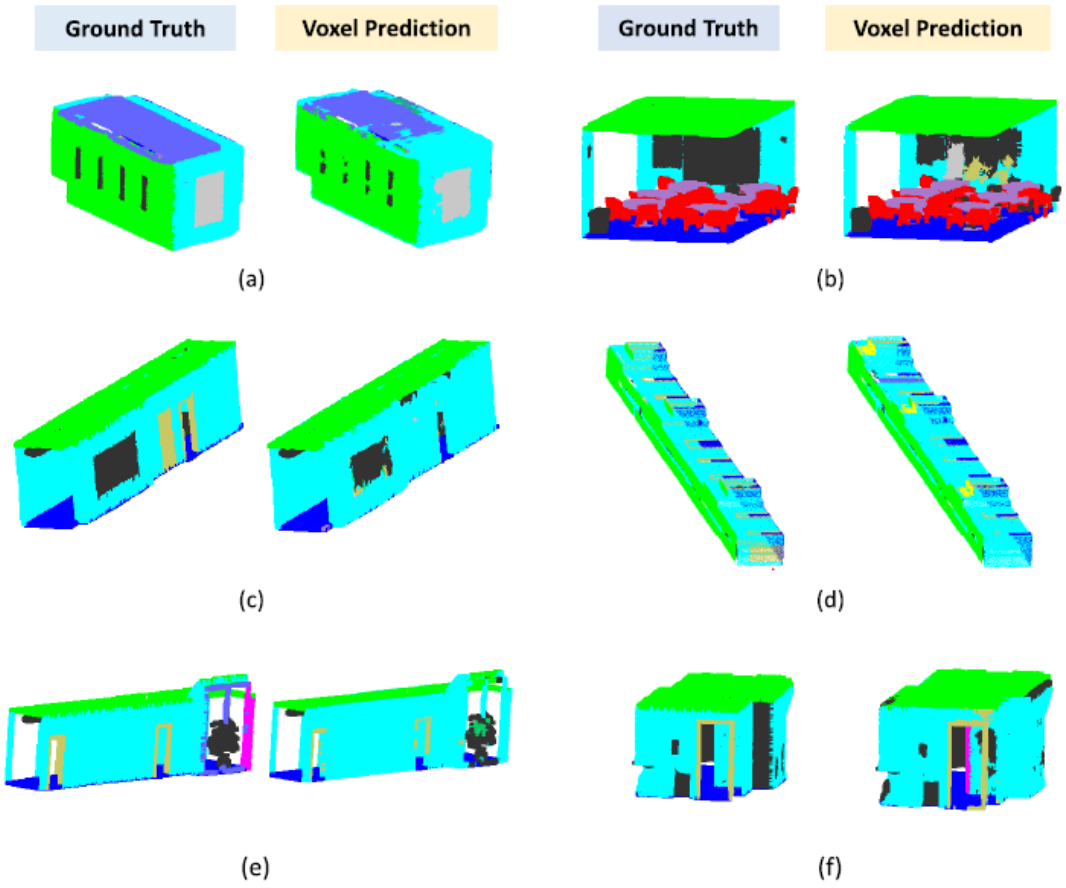

Figure 3.4: Semantic segmentation results of S3DIS.

Semantic Segmentation on Paris-Lille-3D: This data set is composed of three files, including Lille1, Lille2, and Paris, and labeled with ten classes. The first unclassified class will be ignored during training and test. We split the Lille1 data set into two equal folds as Lille1-1 and Lille1-2. The Lille1-1, Lille2, and Paris three folds are treated as training data sets and the Lille1-2 is used as testing data set. To prepare our training data following PointNet (Qi et al., 2017) and PointCNN (Li et al., 2018), we first split the data set along the XOY plane and then sampled them into 5 m × 5 m blocks with a 0.1m buffer area on each side. Points lying in the buffer area are regarded as the contextual information and are not linked to the loss function for model training or class prediction. In addition, points in each block were sampled into a uniform number of 2048 based on the point density and our computation capability.

Table 3.5: OA (%) and mIoU (%) on Paris-Lille-3D data set

	PointNet	PointNet++	DGCNN	Ours
OA	93.9	88.7	96.9	**97.0**
mIoU	40.2	36.1	62.5	**68.2**
Ground	97.5	92.0	98.5	97.9
Building	91.3	79.4	95.2	94.9
Pole	26.5	27.9	57.6	58.8
Bollard	6.3	27.6	52.1	69.8
Trash can	9.0	0.4	42.4	63.8
Barrier	8.5	5.4	35.6	35.9
Pedestrian	4.4	1.8	18.6	38.4
Car	74.3	68.0	93.1	91.7
Natural	44.0	22.3	69.2	62.4

The sampling point in each layer is 2048, 1024, 512, 256, and the number of nearest neighbours is experimentally set to 12. The TGConv is also only applied to the above layers. Due to the limitation on computation capability, we also use the max pooling operation as our aggregation function in TGConv. The other experimental setting is the same as the ScanNet and S3DIS framework. The training epoch is set to 100.

The quantitative evaluations of the experimental results are provided in Table 3.5. In general, our performance is on par with or better than other competitive algorithms, e.g., PointNet (Qi et al., 2017), PointNet++ (Qi et al., 2017), and DGCNN (Wang et al., 2019). Notably, most objects, such as bollard, car, building, and vegetation are fragmented and incomplete due to the mutual occlusion among points. However, our TGNet can still learn to capture their discriminative features for segmentation owing to the powerful structured feature learning capability of TGConv.

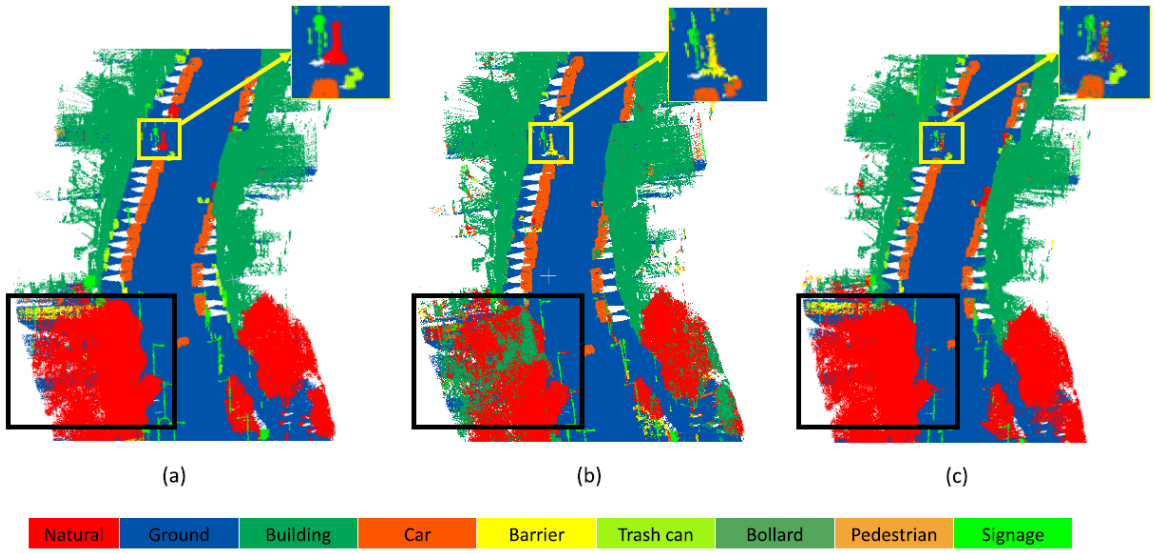

Figure 3.5: Comparison semantic segmentation results of DGCNN and TGNet.

Figure 3.5 shows comparison semantic segmentation results of DGCNN (Wang et al., 2019) and TGNet on Paris-Lille-3D data set. Compared with ground truth, DGCNN and TGNet can segment most points correctly. But there exist some differences between these two results. In the black rectangle, DGCNN misclassified natural points as building points. In our segmentation results, these points were correctly labeled. In the yellow rectangle, there have five objects: signage, natural, car, trash can, and ground. DGCNN classified the natural points as barrier, and trash can as car. These incorrect segmentations did not appear in our TGNet results. Although there have limited natural points predicted wrongly as signage. Thus, we conclude that the exploitation of geometric information in TGNet helps the model to distinguish cluttered objects and shape-similar objects.

3.2.5 Optimizer, Model Size, Memory Usage, and Timing

The proposed method was implemented with Python 3.5 and TensorFlow 1.4 (Abadi et al., 2015) on one GTX 1080ti GPU. We use ADAM optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.001, batch size 12 for the training of our three models. Parameter number and running time are listed in Table 3.6. The model for ScanNet semantic voxel labeling with 4096 input points has 4.32 million parameters for TGNet without CRF layer, and

8.54M parameters for TGNet. TGNet without CRF layer runs 0.058/0.064 s/batch for training/testing, while TGNet runs 0.064/0.068 s/batch for training/testing.

Table 3.6: Parameter number and running time comparisons

Method	# Parameters (million)	Time (second)	
		Training	Testing
TGNet (without CRFasRNN)	4.32	0.058	0.064
TGNet	8.54	0.064	0.068

3.3 Discussion

We have tested our TGNet in both indoor and outdoor environments, where the indoor data were acquired by RGB-D camera and the outdoor data were collected by MLS. The main differences between the above two data sets are the point density and their distribution. Points in indoor scenes are distributed uniformly, while points in outdoor scenes are distributed unevenly and sparsely.

However, the OA (97.0%) in Paris-Lille-3D MLS dataset is much higher than that (88.0%) in S3DIS indoor dataset. We conclude two main reasons for this difference: first, indoor scenes have strong occlusions and tight arrangements of common objects (Qi et al., 2018); second, compared with outdoor scenes, some common objects in indoor scenes have similar shapes and features, thus are hard to differentiate. For example, table and chair, door and window, these object pairs are difficult to distinguish, and they occupy a certain ratio among the whole points. But in outdoor scenes, shape-similar objects are rare. Although, there are sign-like objects (e.g., traffic sign and billboard), they only occupy limited ratio among the whole points.

Based on our experiments, we propose two suggestions when dealing with these two different data sets. For sparse and unevenly distributed MLS data, hierarchically applying convolution in the finest scale points can extract and learn a comprehensive geometric feature. Because geometric information of objects may be severely lost during multiscale sampling. As

for evenly distributed RGB-D data, conducting convolution in multiple scales can exploit both local and global features. This can also reduce the computation cost with guaranteed segmentation performance.

3.4 Chapter Summary

In this chapter, the 3D point cloud segmentation problem was addressed in both indoor and outdoor environments. A novel geometric graph convolution TGConv, which is defined as products of local neighbour point features with geometric features, was proposed. Such geometric features are extracted from local coordinates expressed by a family of Gaussian weighted Taylor kernel functions. This operation can explore the high-level geometric correlations among local neighbours to improve TGConv performance in semantic segmentation. Besides, a parametrized pooling operation, composed of the max and a learnable distanced-based weight function for feature aggregation, were introduced. Based on that, an end-to-end geometric graph convolution architecture TGNet was constructed on the graph representation of point clouds. It employs a multiscale hierarchical architecture by operating TGConv on neighbours at multiple scales and a CRF layer combined within the output layer to further improve the segmentation result.

The experimental results on three different data sets demonstrate that the proposed method achieves 62.2% average accuracy on ScanNet, 57.8% and 68.2% mIoU on S3DIS and Paris-Lille-3D data sets. Quantitative comparison results with several related methods show that our TGNet is more accurate in semantic labeling and has stronger geometric feature expressiveness for 3D point clouds. However, our method still suffers one limitation in multi-object connected area labeling, which is mainly caused by the limited receptive field for TGConv. Thus, how to increase the receptive field and reduce the computation cost will be studied in the future to further improve our algorithm performance.

Chapter 4

3D Indoor Object Detection

This chapter details 3D object detection algorithm in indoor environments. The background of 3D object detection is first introduced in Section 4.1. The proposed framework, including a backbone network for semantic segmentation, a centralization module, and a relation learning module, is then described. The backbone network is constructed based on the algorithm proposed in Chapter 3. Experimental results with corresponding datasets and evaluation metrics are described in Section 4.2. Discussions are presented in Section 4.3. Section 4.4 provides a summary of this chapter.

This chapter is mainly a paper published in a journal and only minor format changes have been made in order to make them to fit into the format of the entire thesis.

4.1 Algorithm Description

The primary problems of 3D object detection are: (1) points distribute sparsely and irregularly (Qi et al., 2017), (2) geometric patterns vary enormously (Ren and Sudderth, 2018), and (3) points locate on the surface of objects, far from their centre (Qi et al., 2019). These challenges lead to the complexity of localization and detection of 3D objects in real scenes. When constructing our detection framework, we face two selections: one-stage detection and two-stage detection. One-stage detection (Yang et al., 2019) generates bounding boxes directly from the extracted point set features without any post-processing steps for refinement. Two-stage detection methods (Hou et al., 2019; Qi et al., 2019; Shi et al., 2019; Yang et al., 2019) mainly consist of two steps: proposal generation and bounding box refinement. One-stage detection is efficient and straightforward but highly relies on the performance of the proposed algorithms. If some difficult objects or geometric-salient objects that could be clearly

distinguished are missed, they have no chance to retrieve (Qi et al., 2019). As for two-stage detection, it considers sufficient possible candidates in the first step and refines the coarse results in the second step. This can sometimes avoid miss detection and thus commonly has higher detection performance and computation cost than the former (Yang et al., 2019). In order to achieve a discriminate performance, we select a two-stage pipeline to construct our model and try to reduce the computation burden.

Different from previous work that inputs RGB-D data as images to 2D CNNs for detection (Gupta et al., 2014), we detect 3D objects from point clouds lifted from depth maps. Geometric attributes and topological structure of 3D objects can be exploited using such data representation (Xu et al., 2018; Li et al., 2020). For example, plane, curve, line, and corner are more easily parameterized and described by 3D learners. In this paper, we introduce an efficient and novel bottom-up two-stage 3D object detection framework from point clouds in indoor scenes, termed geometric relation network (GRNet). We mainly focus on three challenges to improve 3D detection performance:

- Bottom-up feature learning of representative points. Only certain points are selected as candidate points for proposal selection. Their intra-object and inter-object features are exploited.
- Centralization of object surface points. 3D object centres are likely to be empty without any point (Qi et al., 2019). We centralize surface points for more accurate bounding box prediction.
- Object relation learning. Relation features among 3D proposals can attribute to bounding box parameter refinement.

To encode the local geodesic information (e.g., coarse local shape) for representative points, we mimic TGNet (Li et al., 2020) to explore geodesic correlations and attributes among local neighbours. We observe that, in indoor scenes, the topological structure of points in the local region has limited geometric variation. For example, most object surfaces (e.g., beds, desks, and tables) are flat or in regular shape. Thus, we replace the Taylor-Gaussian geometric function with exponentially trilinear interpolation function to approximate local surface

features. We term this new convolution operation as GeoConv. GeoConv is similar to TGConv, but simpler and has fewer parameters.

Our bottom-up backbone framework is constructed based on an encoder-decoder structure, with four-layer down-sampling and two-layer up-sampling. To extract both intra-object and inter-object features, GeoConv is applied to the first two down-sampling layers to exploit the intra-object geometric features. We leverage PointNet (Qi et al., 2017) in the last two down-sampling layers to extract inter-object features. These features are then propagated and concatenated to two up-sampling layers. The output of the backbone network is the selected representative points and their propagated bottom-up features.

Due to the empty object centre, VoteNet (Qi et al., 2019) proposes a Hough voting module to regress the surface points to its centre. Such operation has been proved effective in 3D object detection. However, the scaling problem is not considered, which results in the sub-optimal regression for small or vertical objects. We follow VoteNet (Qi et al., 2019) to propose a centralization module with a scalable loss function. By adding a scaling control parameter in defining centralization loss, object points with a different pattern are centralized in a compact way, which further increases the bounding box prediction results.

Proposals are sampled from these shifted representative points. Their features are learned and aggregated from their nearest neigh- boring points that most are from the same object. Many methods (e.g.,(Yang et al., 2018; Qi et al., 2019; Shi et al., 2019)) predict bounding boxes using such aggregated intra-object features. However, the relation feature between proposals is not exploited. Thus, we propose a simple relation learning module to learn both intra- object and inter-object features to increase the prediction results. Only features from a certain number of nearest neighbours for each proposal are considered for relation feature learning. These neighbours are searched based on the predicted bounding box centre using aggregated intra-object features. Then bounding box parameters are generated as the additive sum of prediction results from relation-based inter-object features and aggregated intra-object features.

4.1.1 Backbone Network

The backbone network is proposed based on the following considerations: (1) intra-object attributes extraction, such as geometric shape, surface variation, and correlation between closed points; (2) inter-object attributes exploitation, e.g., relation features between objects; (3) feature learning and aggregation in a hierarchical way, which can extract point features in different scales; (4) representative points selection, these points are selected to represent the input scene to reduce the computation cost. To meet the above requirements, we construct a bottom-up hierarchical deep framework using a newly defined geometric CNN, GeoConv. The following parts introduce the details of the proposed backbone network.

Although TGNet (Li et al., 2020) proposed the TGConv to explore geodesic correlations and attributes among local neighbours for each point. However, it introduces a high number of parameters. Besides, we observe that, in indoor scenes, most points and their local neighbours lie on planes or regular shape surfaces, which can be described by a simplified parameterized geometric function. To reduce the number of parameters and exploit the geodesic intra-object feature of indoor objects, we propose a new geometric CNN, termed GeoConv. GeoConv is similar to TGConv, but simpler and focuses on regular and simplified geometric characteristics.

Given a 3D point cloud $P = \{p_1, \dots, p_n\} \subseteq R^3$ according to their Euclidean nearest neighbours, a graph $G = (V, E)$ is constructed. $V = \{1, 2, \dots, n\}$ and $E \subseteq V \times V$ denote vertices and edges respectively. The neighbour set for each vertex x is denoted as $y \in \mathrm{N}(x)$. Let $h = \{h_1, h_2, \dots, h_N\}$ be a set of input vertex features, each feature $h_N \in R^F$ corresponds to a graph vertex $i \subseteq V$. F represents each vertex's feature dimension. The output h'_y of GeoConv for each vertex is derived as follows:

$$h'_y = max\left(g_\theta\left(G(\boldsymbol{u}(x,y)) \cdot h_y + h_y\right)\right), \quad y \in \mathrm{N}(x) \tag{4.1}$$

where $G(\cdot)$ is a geometric mapping function: $R^3 \rightarrow R$, which maps the local Euclidean coordinates $\boldsymbol{u}(x,y) = \boldsymbol{u}(y) - \boldsymbol{u}(x)$ between each vertex and its neighbours' Euclidean coordinates to a geometric parameter. Then the product of $G(\boldsymbol{u}(x,y))$ and feature h_y is added

with h_y. $g_\theta(\cdot)$ is the learnable feature mapping function: $R^F \rightarrow R^K$, $max(\cdot)$ denotes the max aggregation function.

As mentioned in TGConv (Li et al., 2020), a family of parametrized Taylor-Gaussian filters were proposed to interpolate arbitrary values at the vertexes of a graph and capture geometric spatial information in a local region. These filters are defined as products of local neighbour point features with geometric features extracted from local coordinates expressed by a family of Gaussian weighted Taylor kernel functions. TGConv is suitable for both indoor and outdoor objects with variable geometric shapes. However, in indoor scenes, common objects have regular geometric shapes. As mentioned in SpiderCNN (Xu et al., 2018), a family of parameterized trilinear interpolation based kernels have been demonstrated to be effective in extracting geometric features. To reduce the number of parameters but also maintain the kernel's expression ability, an exponential-based trilinear interpolation function is used in this paper as the geometric mapping function $G(u(x,y))$ with learnable parameters as:

$$\begin{aligned} G(u(x,y)) &= G(\Delta x, \Delta y, \Delta z) \\ &= e^{(\sigma_0^T + \sigma_1^T \Delta x + \sigma_2^T \Delta y + \sigma_3^T \Delta z + \sigma_4^T \Delta x \Delta y + \sigma_5^T \Delta x \Delta z + \sigma_6^T \Delta y \Delta z + \sigma_7^T \Delta x \Delta y \Delta z)} \end{aligned} \tag{4.2}$$

where $\sigma_i^T (i = 0, \dots, 7)$ is a 1×1 learnable parameter. By varying these parameters, $G(u(x,y))$ can approximate different geodesic values for each vertex x using its neighbour set $y \in \mathrm{N}(x)$.

Because a multi-layer perception (MLP) can approximate an arbitrary continuous function and retains weight sharing as standard convolution (Xu et al., 2018). We use an shared MLP as our feature mapping function $g_\theta(\cdot)$ to map the addition of the original input feature h_y and the products of h_y with a geometric feature $G(u(x,y))$ to a different feature dimension: $R^F \rightarrow R^K$. Max aggregation, which can exploit the most effective features and adaptively explore related neighbour features (Qi et al., 2017), is then applied to aggregate the learned new feature h'_y.

A good backbone framework should meet the above four requirements. In VoteNet (Qi et al., 2019), PointNet++ (Qi et al., 2017) is chosen as the backbone network, which is a hierarchical deep framework with representative point selection. However, the intra-object and

inter-object features are not fully exploited. We construct our backbone framework based on PointNet++, but also explore these two features.

Due to the high density of point clouds in the first two downsampling layers, the extracted local neighbours for each point still construct part of the object surface. As shown in **Figure 4.1**, we apply GeoConv in these two upsampling layers to extract intra-object features. When points are sampled sparsely, especially in the last two encoder layers, geometric attributes (e.g., shape) among extracted neighbours are weakened but the inter-object features (e.g., position or layout) are enhanced. Because using GeoConv in all four encoder layers cannot extract the inter-object features, it sharpens the detection performance. Thus, in the last two downsampling layers, we adopt PointNet (Qi et al., 2017) to extract inter-object features. Then these features are concatenated and interpolated in the following two upsampling layers using PointNet. The output of this backbone is a set of representative points $\{r_i\}_{i=1}^{M}$ where $r_i = [x_i; f_i]$ with $x_i \in R^3$ and $f_i \in R^C$.

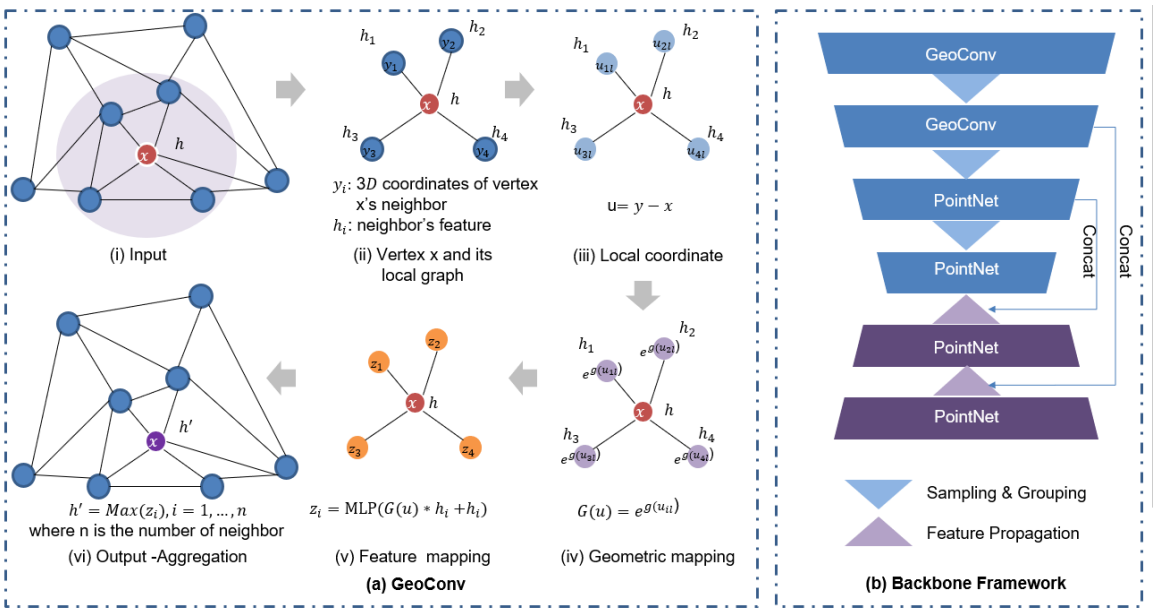

Figure 4.1: Details of our proposed backbone network.

4.1.2 Centralization Module

Due to depth sensors mainly capturing surface points of objects, there are limited points or no points around object centres. Thus, existing point-based networks have a problem in

extracting scene context around the object centre. To solve this, VoteNet (Qi et al., 2019) proposed a Hough voting module to generate new points (votes) that lie close to the object centre. Votes are generated from features of representative points. Then these votes can be grouped and aggregated with a learnable module to generate proposals with enough context information. A vote loss function is introduced to regress the displacements of votes based on the Euclidean distance. This network has been demonstrated to be effective in 3D object detection. However, the scaling problem is not considered in defining their vote loss function. Large objects (i.e., bed) can regress better than small objects (i.e., chair) (Qi et al., 2019). To improve this, we follow VoteNet to construct our centralization module but introduce a scaling control parameter in defining the loss function.

Given a set of representative points $\{r_i\}_{i=1}^{M}$, the centralization module generates offset from each representative point position to its centre independently. This module is composed of a shared MLP module with three fully connected layers, ReLu and batch normalization. The input is the feature $f_i \in R^C$ of representative points, and the output is the 3D position offset $\Delta x_i \in R^3$ in the Euclidean domain and a feature offset $\Delta f_i \in R^C$. Thus, this module generates $c_i = [y_i; g_i]$ from the representative point r_i and has $y_i = x_i + \Delta x_i$ and $g_i = f_i + \Delta f_i$.

The predicted 3D offset Δx_i is supervised by the following loss function:

$$L_{offset-reg} = \frac{1}{N_{pos}} \sum_i \frac{\|\Delta x_i - \Delta x_i^*\|}{\gamma} 1[r_i \; on \; object] \tag{4.3}$$

where $1[r_i \; on \; object]$ represents whether a representative point r_i is on an object surface, N_{pos} is the total number of representative points on object surface. Δx_i^* is the ground truth displacement from the representative point position x_i to the bounding box centre of the object it belongs to. γ is a scale control parameter, which is set to 0.1 in our experiments. Because the offset of different object points varies. Thus, we add a scaling control parameter to enlarge the distance-based regression loss for small objects. Experimental results demonstrate the effective of such scale control parameter.

4.1.3 Proposal Selection and Feature Pooling

The centralization module moves the object surface points to the object centre compactly, while background points still distribute sparsely. Thus, proposal selection should consider such density variation. To ensure the proposal can represent enough possible objects, the sampling and clustering methods are selected according to spatial proximity. A subset of K points are sampled using farthest point sampling (FPS) (Qi et al., 2017) based on the representative point position $\{x_i\}_{\mathrm{i}=1}^{M}$ in 3D Euclidean space. The index of these points is then used to find proposals in shifted representative points $\{y_i\}_{i=1}^{M}$, to get $\{p_k\}_{k=1}^{K}$.

After that, we cluster N group points for each proposal by searching neighbouring points $p_k^{(n)}$ in $\{y_i\}_{i=1}^{M}$, if $\left\|p_k^{(n)} - p_k\right\| \leq r$ for $n = 1, \dots N$. The corresponding feature for each grouped point is denoted as $g_k^{(n)}$. Ball query searching (Qi et al., 2017) is adopted as the nearest neighbour finding method, which only considers neighbouring points in a fixed radius r. N is set to 16 and the r is set to 0.2 according to experimental results. Although smaller radius can include cleaner neighbours (from the same object), it loses context information from background points. Increasing r can contaminate neighbours because more nearby object and clutter points are included.

For each proposal, we use a shared MLP for neighbouring points' feature mapping. The max operation is applied for feature aggregation:

$$F_k = \underset{n = 1, \dots N}{max}\{MLP([r_k^{(n)}; g_k^{(n)}])\} \tag{4.4}$$

where $r_k^{(n)} = p_k^{(n)} - p_k$ is the relative coordinate between neighbouring points to its proposal, and $F_k \in R^{(3+C)}$. This aggregated output feature represents the intra-object attribute, because neighbouring points mainly come from the same object.

4.1.4 Object Relation Learning module

As for discriminate 3D object detection, intra-object feature and inter-object feature are of the same importance. The above aggregated proposal feature represents the intra-object feature generated from points that lies on the same object surface. However, in the real scene, there exists relationships between objects. To leverage the inter-object feature between co-

occurrence and locations of objects for better reasoning, we propose an object relation learning module.

We only consider S nearest neighbouring proposals for each proposal to leverage their relation features. These neighbouring proposals are searched based on the predicted bounding box centre position. In this paper, a 3D bounding box is represented as $(x, y, z, h, w, l, \theta)$, where (x, y, z) is the object centre coordinates, (h, w, l) is the object size (height, width, length), and θ is the object orientation. Three fully connected layers are applied to predict bounding box parameters $B_{k,1}(x_{k,1}, y_{k,1}, z_{k,1}, h_{k,1}, w_{k,1}, l_{k,1}, \theta_{k,1})$ using the intra-object feature F_k.

Each proposal neighbours are searched using the predicted bounding box centre position $(x_{k,1}, y_{k,1}, z_{k,1})$. We formulate the relation between a proposal to its neighbouring proposals as a region-to-region undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, S\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denote vertices and edges respectively. The ith neighbouring proposal feature is denoted as $F_{i,k}$. We then seek to learn the relation parameter $\alpha_{i,k} \in R^{1\times(3+C)}$ $(i = 1, .. S)$, and the object relation feature F_{rk} as follows:

$$\alpha_{i,k} = \frac{\exp\left(\widetilde{\alpha_{i,k}} * F_{i,k}\right)}{\sum_{i=1}^{S} \exp\left(\widetilde{\alpha_{i,k}} * F_{i,k}\right)} \tag{4.5}$$

$$F_{rk} = \sum_{i=1}^{S} \alpha_{i,k} * F_{i,k} + F_k \tag{4.6}$$

where $\widetilde{\alpha_{i,k}}$ $(i = 1, .. S)$ is a $1 \times (3 + C)$ learnable parameter. This newly generated relation feature F_{rk} is then sent to three fully connected layers for bounding box parameter prediction, which is denoted as $B_{k,2}(x_{k,2}, y_{k,2}, z_{k,2}, h_{k,2}, w_{k,2}, l_{k,2}, \theta_{k,2})$. The final output of this network is the additive sum of $B_{k,1}$ and $B_{k,2}$. **Figure 4.2** illustrates the framework of this module.

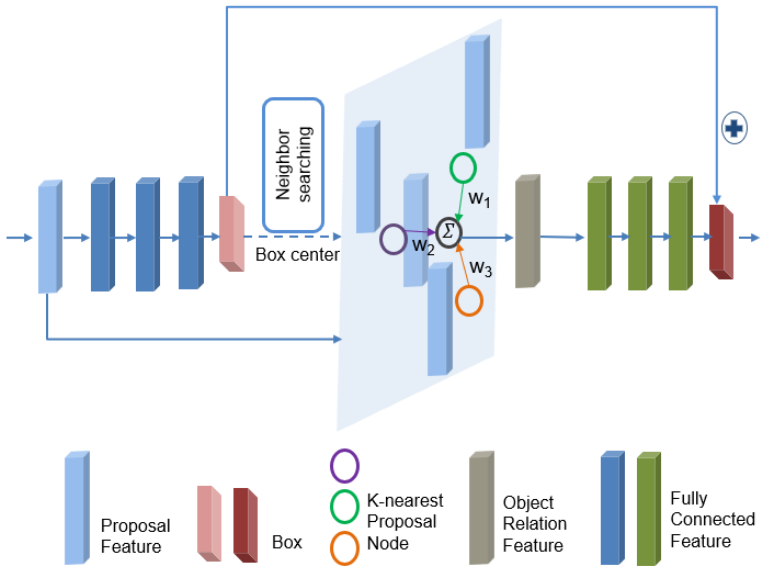

Figure 4.2: The framework of the object relation learning module

Following VoteNet (Qi et al., 2019), we use a hybrid of classification and regression formulation. For angle prediction, we pre-define N_a as equally split angle bins and classify the proposal angle into different bins. Residual is regressed with respect to the bin value. N_a is set to 12 in our experiments. Finally, the non-maximum suppression (NMS) based on the objectness score and semantic classification score is applied to eliminate redundant proposals. Specifically, we keep up to 256 proposals during training and testing.

4.1.5 Loss Function

To optimize the proposed end-to-end framework, a multi-task loss is applied. It includes a centralization loss, a 3D bounding box estimation loss, a semantic classification loss, and an objectness loss:

$$L_{GRNet} = L_{offset-reg} + \lambda_1 L_{box} + \lambda_2 L_{sem-cls} + \lambda_3 L_{obj-cls} \quad (4.7)$$

where $\lambda_1 = 1$, $\lambda_2 = 0.1$ and $\lambda_3 = 0.5$. These parameters are used to weight the losses to maintain that they have similar scales.

$L_{offset-reg}$ is as defined in Section 4.2. As for the last three losses, we follow VoteNet (Qi et al., 2019) to construct them. Both the objectness loss and the semantic classification loss

are cross-entropy loss, but for two classes and C semantic classes, respectively. Only positive proposals are considered in calculating the box and semantic losses, which are normalized by the total number of positive proposals. Those proposals, whose distances to their nearest ground truth centre are less than 0.2m, are defined as positive proposals. For those proposals with distance larger than 0.5m are denoted as negative proposals. Those proposals whose distances are between these two thresholds are neglected. These distance thresholds are determined by experimental results.

The box loss is composed of the centre regression, heading estimation and size estimation sub-losses using L1-smooth loss (Qi et al., 2018):

$$L_{box} = L_{center-reg} + 0.1L_{ang-cls} + L_{angle-reg} + 0.1L_{size-cls} + L_{size-reg} \quad (4.8)$$

where centre regression loss $L_{center-reg}$ is defined by Chamfer loss (Fan et al., 2017).

4.2 Experiments

4.2.1 Experimental Setup and Implementation

Dataset. The performance of our method is evaluated on two indoor datasets: SUN-RGBD (Song et al., 2015) and ScanNetV2 (Dai et al., 2017). SUN-RGBD is collected using multiple different RGB-D cameras with varying resolutions from different indoor scenes. It contains 5,285 training images and 5,050 testing images, respectively. There are 37 object categories labeled with amodal oriented 3D bounding boxes. We report model performance on the testing set. Point cloud data are acquired following the method provided by VoteNet (Qi et al., 2019). Detection results on the 10 most common categories are reported.

ScanNetV2 contains 1,201/312 training/testing RGB-D images collected from various indoor rooms. These scenes are labeled with 18 object classes for semantic segmentation and instance segmentation. Compared with SUN-RGBD dataset, scenes in this dataset are annotated with more categories and cover larger areas. Point clouds are sampled from the reconstructed meshes. Because the orientation of the bounding box is not annotated, the axis-aligned bounding boxes are predicted, as in VoteNet (Qi et al., 2019).

Evaluation Criteria. Following VoteNet (Qi et al., 2019) and 3D-BoNet (Yang et al., 2019), the average precision metric AP_{3D} of 3D detection results is adopted as our evaluation

criteria. The predicted bounding box B_p is treated as a valid detection result only its 3D overlap area (IoU) between the predicted bounding box B_p and the ground truth bounding box B_{gt} exceeds a certain ratio. IoU is calculated using the following evaluation metric:

$$IoU = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \tag{4.9}$$

Predicted bounding boxes with 3D IoU results exceeding 0.25 and 0.5 are used to evaluate the detection performance for all classes in both two datasets.

Implementation Details. In our experiments, we implement our model based on VoteNet (Qi et al., 2019), an open-source framework for 3D object detection built on the PyTorch platform. This framework is composed of three-part: backbone network, Hough voting module, and object proposal and classification module. The backbone network is based on PointNet++ (Qi et al., 2017), which has several set-abstraction (SA) layers and feature propagation (FP) layers with skip connections. In the first two SA modules, we replace the PointNet (Qi et al., 2017) with our proposed GeoConv. Our centralization module is similar to the Hough voting module, but we replace the vote loss function with our proposed scalable loss function. The last module is also replaced by our object relation learning module for discriminate object bounding box reasoning and refining. The training epoch is set to 200.

The general setting of our backbone network for these two datasets are listed in **Table 4.1**. The input number of points, sampling radius, the number of nearest neighbours, and the mlp output sizes to each layer are introduced. Most hyper-parameters in the same layer of two datasets are similar, only limited parameters are different. Because SUN-RGBD has more sparse point density than ScanNetV2, the sampling radius in SUN-RGBD is reduced to 0.1 and 0.2 in the first two SA layers with the same number of nearest neighbours as 32. Such changes can ensure that the GeoConv in the first two layers extract enough intra-object geometric information in both two datasets.

Table 4.1: General setting of the backbone network on ScanNetV2 and SUN-RGBD datasets

Layer	Backbone (Dataset)	#Point	Sampling radius (m)	#neighbours	mlp
SA1 (GeoConv)	ScanNetV2	2048	0.2	64	[4,64,64,128]
	SUN-RGBD	2048	0.1	32	[4,64,64,128]
SA2 (GeoConv)	ScanNetV2	1024	0.4	32	[128,128,128,256]
	SUN-RGBD	1024	0.2	32	[128,128,128,256]
SA3 (PointNet)	ScanNetV2	512	0.8	16	[256,128,128,256]
	SUN-RGBD	512	0.8	16	[256,128,128,256]
SA4 (PointNet)	ScanNetV2	256	1.2	16	[256,128,128,256]
	SUN-RGBD	256	1.2	16	[256,128,128,256]
FP1 (PointNet)	ScanNetV2	512	-	3	[512,256,256]
	SUN-RGBD	512	-	3	[512,256,256]
FP2 (PointNet)	ScanNetV2	1024	-	3	[512,256,256]
	SUN-RGBD	1024	-	3	[512,256,256]

4.2.2 Ablation Studies

To demonstrate the effectiveness and importance of each proposed individual module, some ablation studies were conducted on both SUN-RGBD and ScanNetV2 datasets. When testing each module, the remaining modules kept unchanged. The followings are the detailed evaluation of these modules.

(1) Contribution of GeoConv in backbone network

As mentioned in the Section 5.1.2, the backbone network is based on PointNet++ (Qi et al., 2017), which has several SA modules and FP modules with skip connections and PointNet (Qi et al., 2017) for feature mapping. We replace PointNet in some SA modules with our proposed GeoConv to extract geometric intra-object features for representative points. When testing the effectiveness of GeoConv, the scaling parameter of centralization loss for ScanNetV2 was set to 0.1 and SUN-RGBD was set to 0.2, and the neighbouring number in the object relation learning module for both two datasets was set to 3. We found that, as shown in **Table 4.2**, the highest performances for both two datasets were achieved when the PointNet in

the first two SA modules was replaced by GeoConv while keeping others unchanged. Because the GeoConv is mainly focused on the intra-object geometric features learning, with an increased sampling ratio, the relation features between those remaining points are increasing. The geometric attributes among these points are weakened. Thus, the performance dropped when replacing more PointNet layers among SA modules with the GeoConv layer.

Table 4.2: Contribution of GeoConv in backbone network on ScanNetV2 and SUN-RGBD datasets

	SA1	SA2	SA3	SA4	FP1	FP2	mAP@0.25 (%)		mAP@0.5 (%)	
							ScanNetV2	SUN-RGBD	ScanNetV2	SUN-RGBD
PointNet++ (Qi et al., 2019)	PT	PT	PT	PT	PT	PT	58.3	57.2	37.8	33.9
#1GeoConv-PointNet++	GC	PT	PT	PT	PT	PT	57.6	57.5	36.7	33.7
#2GeoConv-PointNet++	GC	GC	PT	PT	PT	PT	**59.1**	**58.4**	**39.1**	**34.9**
#3GeoConv-PointNet++	GC	GC	GC	PT	PT	PT	58.6	57.4	37.7	33.5
#4GeoConv-PointNet++	GC	GC	GC	GC	PT	PT	57.7	56.6	36.9	32.9

Note: #GeoConv-PointNet++: represents the number of PointNet in PointNet++ replaced by our proposed GeoConv in SA modules. PT represents PointNet (Qi et al., 2017), GC means GeoConv.

(2) Comparison of different scaling parameters

In this part, we tested different scaling parameters to see their effectiveness. We selected 0.05, 0.1, 0.15, 0.2 and 0.25 in our experiments, as shown in **Table 4.3**. The highest results for ScanNetV2 with 59.1% mAP@0.25 and 39.1% mAP@0.5 were achieved using 0.1, while the best results for SUN-RGBD were accomplished with 58.4% mAP@0.25 and 34.9% mAP@0.5

using 0.2. Because SUN-RGBD has more sparse point density than ScanNetV2, the best scaling parameter for SUN-RGBD was 0.2. The performances for these two datasets with larger or smaller scaling parameters than the parameters with the best results were decreased. The reduced scaling parameter leads to a compact grouping, which causes the contamination of non-object points in proposal feature pooling. With a larger scaling parameter, the aggregated intra-object feature cannot consume enough effective neighbouring features. Thus, detection results decreased.

Table 4.3: Effectiveness of different scaling parameters on ScanNetV2 and SUN-RGBD datasets

Scaling parameter	mAP@0.25 (%)		mAP@0.5 (%)	
	ScanNetV2	SUN-RGBD	ScanNetV2	SUN-RGBD
0.05	57.0	56.7	38.0	34.3
0.1	**59.1**	57.3	**39.1**	33.7
0.15	57.9	56.9	38.2	34.3
0.2	58.7	**58.4**	38.6	**34.9**
0.25	58.1	56.6	37.5	33.3

(3) Effectiveness of Object Relation Learning Module

We also tested the contribution of our proposed object relation learning module on ScanNetV2 and SUN-RGBD datasets. As shown in **Table 4.4**, without (w/o) the relation learning module, the detection results dropped 1.4% at mAP@0.25 and 1.7% mAP@0.5 on ScanNetV2 and decreased 0.7% mAP@0.25 and 1.8% mAP@0.5 on SUN-RGBD, compared to their best results. Relation learning from 3 nearest neighbour proposals achieved the best results with 59.1% mAP@0.25 and 39.1% mAP@0.5. An increasing number of neighbouring proposals may induce more irrelevant features for bounding box reasoning. Thus, the detection performance was weakened. With a reduced number of neighbouring proposals, e.g., 2

neighbours, some important relation features are missing. This results in a decreased performance.

Table 4.4: Effectiveness of Object relation learning module on ScanNetV2 and SUN-RGBD datasets

Relation module	mAP@0.25 (%)		mAP@0.5 (%)	
	ScanNetV2	SUN-RGBD	ScanNetV2	SUN-RGBD
w/2nn	57.1	57.4	36.5	34.1
w/3nn	**59.1**	**58.4**	**39.1**	**34.9**
w/4nn	56.8	57.0	36.6	34.4
w/5nn	57.9	56.9	38.3	32.8
w/6nn	57.5	56.7	38.0	31.9
w/o	57.8	57.7	37.3	33.5

4.2.3 Object Detection Results

(1) ScanNetV2 Detection Results

Quantitative detection results of ScanNetV2 are listed in **Table 4.5**. GRNet outperforms all previous methods, e.g., 3DSIS Geo (Hou et al., 2019), 3DSIS 5views (Hou et al., 2019), and VoteNet (Qi et al., 2019) by at least 0.5% mAP@0.25 and 5.5% mAP@0.5 increases. The important improvement mainly comes from mAP@0.5 results. Compared with VoteNet (Qi et al., 2019), our method improves the previous state of the art by more than 20.0% AP in the category “counter”, 11.0% AP in “desk”, 10.0% AP in “bookshelf”, 7.0% AP in 3 categories such as sink, and 4.0% AP in the other 8 categories. As illustrated in the ablation studies, the centralization module centralized the surface points in a compact way, which contributes to a more effective proposal feature aggregation. The object relation learning module extracted the useful nearest neighbours feature for better bounding box reasoning. These two modules improve the detection results for mAP@0.5. As for the results at mAP@0.25, GeoConv

improves the performance of the representative points' feature by considering both intra-object and inter-object features. **Figure 4.3** shows some examples of the detection result. Small and shape-similar objects are easy to be mis-detected. There also exists wrong detection in density-compact areas, e.g., corners.

Table 4.5: 3D object detection scores per category on the ScanNetV2 (validation) dataset

	3DSIS Geo	3DSIS 5views	VoteNet	GRNet (Ours)	3DSIS Geo	3DSIS 5views	VoteNet	GRNet (Ours)
	mAP@0.25 (%)				mAP@0.5 (%)			
Cab	19.8	12.8	36.3	**39.5**	5.1	5.7	8.1	**9.8**
Bed	69.7	63.1	87.9	**88.8**	42.2	50.3	76.1	**80.3**
Chair	66.2	66.0	88.7	**89.2**	50.1	52.6	67.2	**71.0**
Sofa	71.8	46.3	**89.6**	88.3	31.8	55.4	68.8	**76.0**
Tabl	36.1	26.9	**58.8**	58.2	15.1	22.0	42.4	**44.6**
Door	30.6	8.0	47.3	**48.5**	1.4	10.9	15.3	**20.6**
Wind	10.9	2.8	**38.1**	32.7	0.0	0.0	6.4	**8.9**
Bkshf	27.3	2.3	44.6	**47.0**	1.4	13.2	28.0	**38.2**
Pic	0.0	0.0	**7.8**	4.9	0.0	0.0	**1.3**	1.2
Cntr	10.0	6.9	56.1	**63.5**	0.0	0.0	9.5	**29.7**
Desk	46.9	33.3	**71.7**	69.8	13.7	23.6	37.5	**49.0**
Curt	14.1	2.5	47.2	**48.5**	0.0	2.6	11.6	**18.4**
Fridg	**53.8**	10.4	45.4	49.1	2.6	24.5	27.8	**34.2**
Showr	36.0	12.2	57.1	**66.4**	3.0	0.8	10.0	**13.4**
Toil	87.6	74.5	**94.9**	94.1	56.8	71.8	86.5	**90.1**
Sink	43.0	22.9	**54.7**	49.7	8.7	8.9	16.8	**20.9**
Bath	84.3	58.7	**92.1**	90.9	28.5	56.4	78.9	**82.6**
Ofurn	16.2	7.1	**37.2**	35.6	2.6	6.9	11.5	**15.5**
mAP	40.2	25.4	58.7	**59.1**	14.6	22.5	33.5	**39.1**

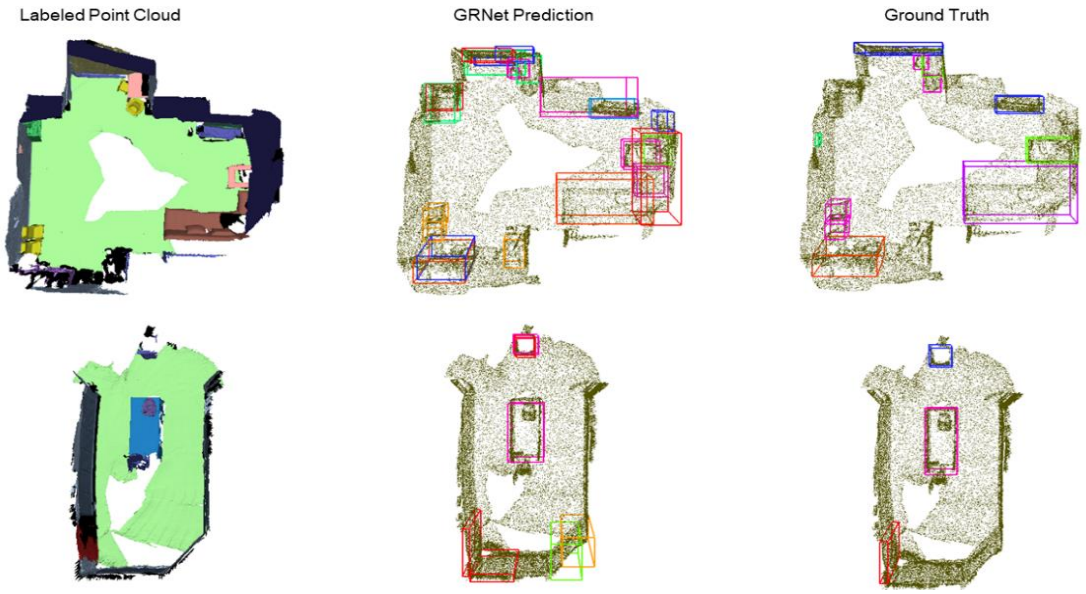

Figure 4.3: Qualitative results of 3D object detection in ScanNetV2.

(2) SUN-RGBD Detection Results

Quantitative results in Table 4.6 illustrates the detection performance for all classes on SUN-RGBD dataset. GRNet outperforms all previous methods by at least 0.7% mAP@0.25 increase and 2.8% mAP@0.5 increase in SUN-RGBD with point clouds input only. Compared with other detection performances. e.g., DSS (Song and Xiao, 2016), COG (Ren and Sudderth, 2016), 2D-driven (Lahoud and Ghanem, 2017), F-PointNet (Qi et al., 2018), PointFusion (Xu et al., 2018), and VoteNet (Qi et al., 2019), our algorithm can achieve the state-of-art or on-par-with mAP@0.25 detection results on large and geometric-salient objects, such as bed, sofa, bathtub, table and chair. For geometric-weak objects, such as picture and dresser, the improvements are limited. As for detection results on mAP@0.5, our algorithm outperforms the VoteNet (Qi et al., 2019) on 8 categories and on-par-with it on 2 categories. As shown in **Figure 4.4**, the large object with enough scanned point clouds, such as beds, can be detected accurately. However, for thin and density-sparse objects (e.g., bookshelves, desks, and dressers), misdetection occurs commonly. Besides, for shape similar objects, such as tables and nightstands, they are easy to be mis-predicted.

Table 4.6: 3D object detection scores per category on the SUN-RGBD (test) dataset

(use one decimal place in this table)

	DSS	COG	2D-driven	F-PointNet	Point-Fusion	VoteNet	GRNet (ours)	VoteNet (Baseline)	GRNet (ours)
	mAP@0.25 (%)							mAP@0.5 (%)	
Input	Geo& RGB	Geo& RGB	Geo& RGB	Geo& RGB	Geo& RGB	Geo only	Geo only	Geo only	Geo only
Bathtub	44.2	58.3	43.5	43.3	37.3	74.4	**76.8**	41.4	41.3
Bed	78.8	63.7	64.5	81.1	68.6	83.0	**84.3**	49.5	**54.9**
Bookshelf	11.9	31.8	31.4	33.3	**37.7**	28.8	29.3	5.4	5.0
Chair	61.2	62.2	48.3	64.2	55.1	75.3	**76.2**	52.3	**55.9**
Desk	20.5	45.2	**27.9**	24.7	17.2	22.0	26.0	4.9	**5.8**
Dresser	6.4	15.5	25.9	**32.0**	24.0	29.8	26.1	12.1	**14.9**
Night-Stand	15.4	27.4	41.9	58.1	32.3	62.2	59.2	33.9	**36.1**
Sofa	53.5	51.0	50.4	61.1	53.8	64.0	**64.8**	42.9	**46.1**
Table	50.3	**51.3**	37.0	51.1	31.0	47.3	51.1	18.5	**24.6**
Toilet	78.9	70.1	80.4	**90.9**	83.8	90.1	90.4	60.5	**63.9**
mAP	42.1	47.6	45.1	54.0	45.4	57.7	**58.4**	32.1	**34.9**

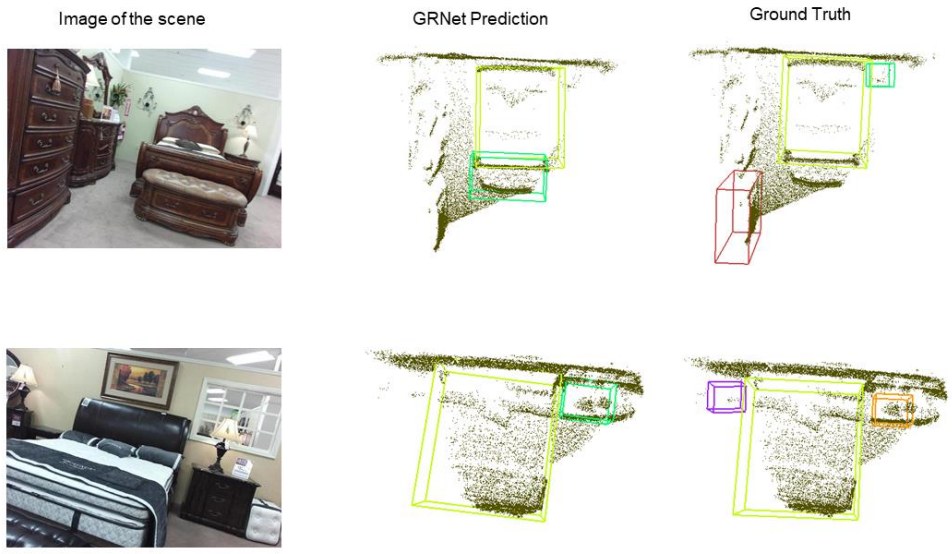

Figure 4.4: Qualitative results on SUN-RGBD.

4.2.4 Optimizer, Model size, Memory Usage and Timing

We implemented our model with Python 3.5 and PyTorch 1.0 on one GTX 1080ti GPU. ADAM optimizer (Kingma and Ba, 2014), with an initial learning rate of 0.001, was adopted. The learning rate was decayed at 80, 120, 160 epochs, respectively, with a 0.1 decay rate. The batch size was set to 8 for both training and testing our GRNet-SUN-RGBD and GRNet-ScanNetV2 models. As shown in **Table 4.7**, the model for GRNet- SUN-RGBD with 20000 input points has 13.5M parameters and 17.8M parameters for GRNet-ScanNetV2 with 40,000 input points. GRNet-SUN-RGBD runs 0.12 seconds per frame or scan for training, while GRNet-ScanNetV2 runs 0.10 seconds per frame or scan for training. Because the GRNet (ScanNetV2) has larger model size than VoteNet, its computation cost increases. However, as for GRNet (SUN-RGBD), although it increases around 2MB model size compared to VoteNet, their computation costs are the same. The main reason is that the GRNet (SUN-RGBD) reduces the search radius and sampling neighbours in the first two SA modules in the backbone network. Such reduction largely relieves the computation burden.

Table 4.7: Model size and processing time (per frame or scan)

Method	Model size	SUN-RGBD	ScanNetv2
F-PointNet	47.0MB	0.09s	-
3D-SIS	19.7MB	-	2.85s
VoteNet	11.2MB	0.10s	0.14s
GRNet (ScanNetV2)	17.8MB	-	0.22s
GRNet (SUN-RGBD)	13.5MB	0.10s	-

4.3 Discussion

We have tested our GRNet in two indoor environments, which show some differences in point density, room layout, and area. SUN-RGBD has a larger room area, sparser point density, and less labeled objects compared with ScanNetV2. Thus, the application of GeoConv should consider such differences. The sampling radius of GeoConv in the first two SA modules is 0.1 and 0.2 in SUN-RGBD, 0.2 and 0.4 in ScanNetV2, respectively.

In addition, the scaling parameter is also different. Labeled objects in ScanNetV2 are smaller and more compact than SUN-RGBD. As mentioned in VoteNet, voting is only useful for points that are far away from the object centre (Qi et al., 2019). Thus, in order to improve the centralization results for small objects, 0.1 scaling parameter was applied as the scaling parameter. However, in SUN-RGBD dataset, labeled objects are larger than ScanNetV2, the best detection result was achieved using 0.2 scaling parameter. We also found that the scaling parameter and relation learning module are more effective in predicting mAP@0.5 bounding box parameters. A compact centralization attributes to neighbours' inter-object and intra-object features learning, which results in a more accurate bounding box prediction. The subgraph (a) in **Figure 4.5** shows the centralization effects. The further improvement for both mAP@0.25 and mAP@0.5 should consider the RGB information, especially for geometric-weak objects, such as the picture.

Finally, as shown in **Figure 4.5**, those proposals can cover all the labeled objects. However, the post-processing by using NMS based on the objectness score and semantic classification score removed low confident proposals (which were actually true positive proposals). Thus,

the final detection results missed the true bounding box for objects. From the subgraph (f) in **Figure 4.5**, we can see that the right nightstand is not detected. Additionally, the position of confident proposals affects the predicted bounding box position, which can be seen in the subgraph I and (f) in **Figure 4.5**. Thus, how to associate the objectness score with the accuracy of the predicted bounding box should be studied in the future to improve our final detection results.

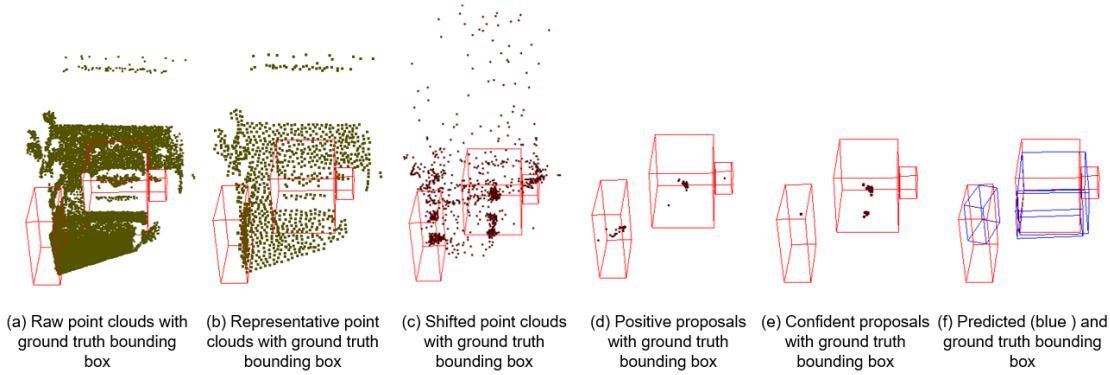

Figure 4.5 Staged outputs of GRNet.

4.4 Chapter Summary

In this chapter, an end-to-end point cloud geometric relation network (GRNet) focused on 3D object detection in indoor scenes was proposed. The oriented 3D bounding boxes (i.e., centre, heading angle, and size) and semantic classes of objects were estimated. This network can exploit both intra-object and inter-object features in a bottom-up hierarchical way using the proposed backbone network for representative points. Then, a centralization module with a scalable loss function was introduced to centralize object points to its centre. Proposal points were sampled from these shifted representative points, following a proposal feature pooling operation. Finally, an object-relation learning module was applied to predict bounding box parameters. Such parameters are the additive sum of prediction results from relation-based inter-object features and aggregated intra-object features.

This model achieves state-of-the-art 3D detection results with 59.1% mAP@0.25 and 39.1% mAP@0.5 on ScanNetV2 dataset, 58.5% mAP@0.25 and 34.1% mAP@0.5 on SUN-

RGBD dataset. Quantitative comparison performance and qualitative results demonstrated the effectiveness of our proposed framework in 3D object detection. However, RGB features are not exploited in this paper, which may contribute to a further improvement for geometric-weak objects. Besides, how to associate the objectness score with the accuracy of the predicted bounding box should be studied in the future to improve the performance of our method.

Chapter 5

2D-Driven 3D Object Detection from Indoor and Outdoor Environments

In this chapter, the framework of the proposed 2D-driven 3D object detection algorithm is introduced. In Section 5.1, the introduction for the 2D-driven 3D model and the implementation details of the proposed algorithm are provided. In Section 5.2, the details for each sub-network, datasets, evaluation metrics, and experimental results are provided. Section 5.3 discusses the experimental results of the proposed framework. Section 3.4 concludes this chapter.

This chapter is mainly a manuscript submitted to a journal and only minor format changes have been made in order to make them to fit into the format of the entire thesis. © [2020] IEEE. Reprinted, with permission, from [Li, Y., Ma, L., Tan, W., Sun, C., Cao, D., Li, J. 2020. 2020. 3D Object Detection from Indoor and Outdoor Frustum Point Clouds, IEEE Transactions on Intelligent Transportation Systems, submitted.]

5.1 Algorithm Description

3D object detection is crucial in many applications, such as autonomous driving (Gao et al., 2018), modeling (Zhong et al., 2018), computer vision (Mousavian et al., 2017), and remote sensing (Luo et al., 2019). 3D data can be obtained by LiDAR or RGB-D cameras. Thus, some 3D data have accompanied corresponding 2D images. There are multiple ways to extract 3D objects from these data, e.g., point-based (Qi et al., 2019; Shi et al., 2019; Li et al., 2020), view-based (Chen et al., 2017; Ku et al., 2018), and multi-sensor fusion-based (Liang et al., 2018; Wang and Jia, 2019) methods. The point-based scheme detects 3D objects directly from point clouds, the view-based scheme coverts the 3D points into 2D views and leverages the mature 2D detector to detect objects, while the multi-sensor fusion scheme explores object features from 2D and 3D data together. 3D point clouds are irregular and sparse, locating

objects accurately is hard to achieve. To leverage the mature 2D detectors and high-resolution images, following F-PointNet (Qi et al., 2018), we make effective use of 2D images and 3D detection schemes to enhance the 3D detection performance.

Intuitively, we detect 2D object proposals in the input images. A 2D detected proposal box can be lifted to a 3D frustum area using a known camera-LiDAR projection matrix. Points in this frustum space are collected as inputs to the point-based detection framework to predict the amodal bounding box. However, there are two main considerations when detecting objects in frustum point clouds:

- How to detect the object accurately from the frustum points with background and clutter disturbance.
- How to improve the incorrect detection results caused by the inaccurate 2D proposal boxes. The detected 2D proposal boxes cannot bound the object instances precisely.

To solve the above challenges, our detection framework is composed of the following two stages: bounding box prediction and bounding box refinement. Stage-1 networks predict the amodal bounding box from the frustum point clouds. To compensate the incorrect 2D detection results, stage-2 networks refine the predicted bounding box using points in the enlarged predicted bounding box. Both two stage networks contain point cloud segmentation, residual centre prediction and bounding box prediction modules.

For stage-1 detection networks, the primary challenge comes from the background and clutter point disturbance. To improve the detection accuracy, foreground object points are extracted after the point cloud segmentation module. Bounding box parameters are predicted from these foreground points. As mentioned in (Shi et al., 2019), the context information of the predicted objects can improve the detection results. Thus, in this paper, a context point extraction method is proposed to extract the context points from background points. The context and the foreground points are combined as the context foreground points for further bounding box prediction.

For stage-2 refinement networks, points in the enlarged predicted bounding box are collected as inputs for bounding box refinement. The frustum space lifted from 2D proposal boxes is not applied in this stage for box point collection. This operation can extract object-

specific points with useful context points compared to the frustum points. The detection pipeline in the first stage is applied again in this stage for amodal bounding box refinement. However, the context point extraction module is removed because the collected input points in this stage can be viewed as the context foreground points.

In order to describe the detected object comprehensively in both two stage networks, the global feature, which represents the surrounding background of the object, and the local feature that describes the object attributes, should be leveraged. In our paper, the global feature learned in the point cloud segmentation module is used as the global context feature, while the global feature obtained from the context foreground points is viewed as the object-specific feature. These two features are concatenated with semantic cues extracted from the 2D proposals for bounding box parameter prediction. Figure 5.1 shows the detection framework of our proposed method.

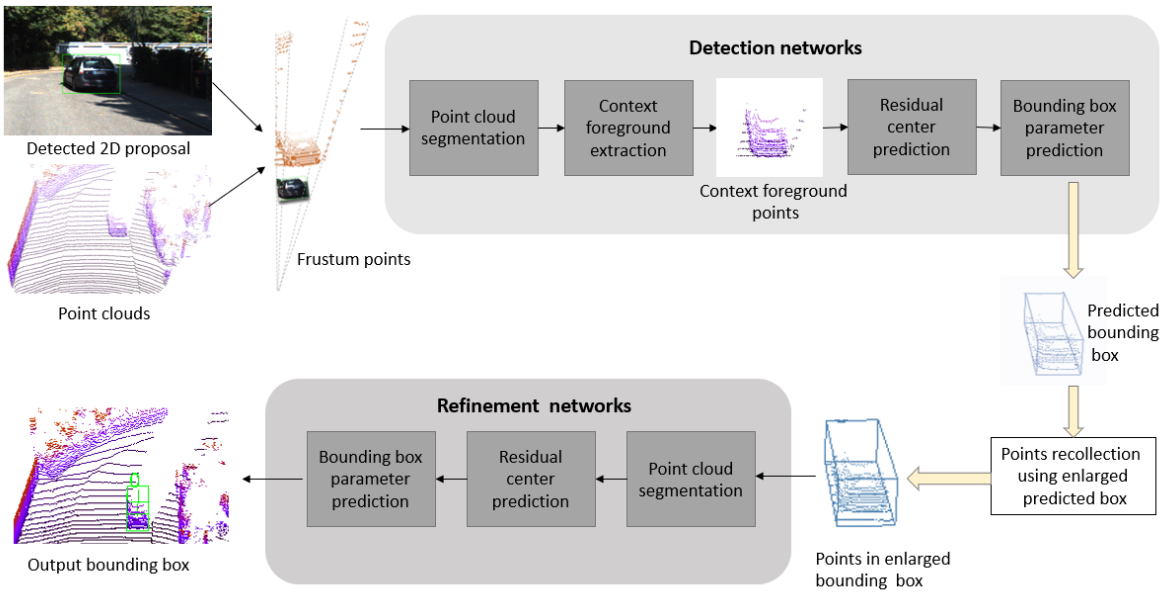

Figure 5.1: 3D object detection framework.

5.1.1 Context Foreground Point Segmentation

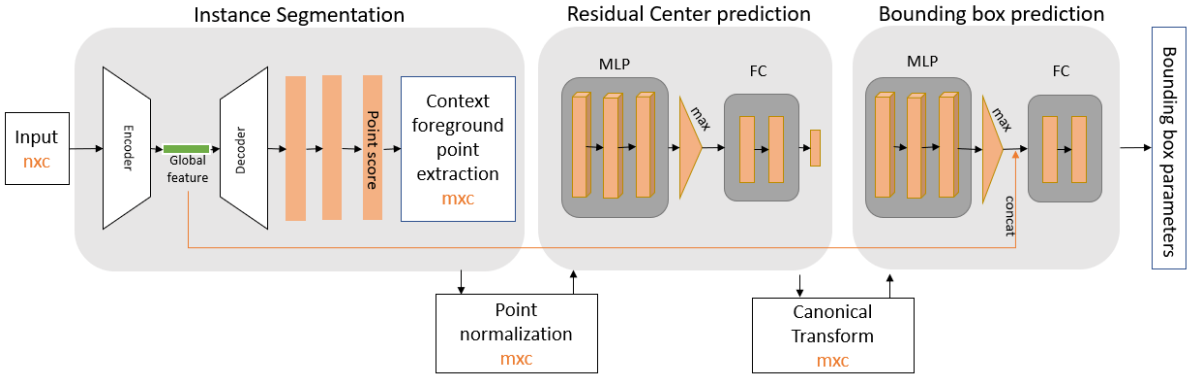

Figure 5.2: Detection networks.

Following F-PointNet (Qi et al., 2018), in our method, we assume that the point clouds have corresponding images. Instead of detecting the objects directly from point clouds, objects in 2D images are first detected with 2D bounding boxes. Then these boxes are lifted to frustums using the known camera projection matrix. Such frustums define the 3D searching area for object detection. Points in the frustum are collected to form a frustum point cloud. This mechanism can leverage the mature 2D detectors and largely reduce the computation cost for 3D object detection. To improve the rotation-invariance of the frustums, these frustums are normalized to make the centre axis of the frustum orthogonal to the image plane.

Within the normalized frustum points, there are two pipelines to detect the amodal object:1) directly detect the object from the point clouds; 2) extract foreground points first and then predict the bounding box using these points (Qi et al., 2018). Although the frustum points reduce the most non-relevant backgrounds and clutter, the remaining points and overlap objects still disturb the precise localization of the amodal object. Although the first pipeline is simple, to ensure the detection performance, we follow F-PointNet (Qi et al., 2018) to construct our detection framework. Figure 5.2 shows the detection pipeline.

The foreground point segmentation can locate the associated object accurately with foreground context. To exploit the geometric features for each foreground point, we apply multi-scale GeoConv (Li et al., 2020) with the encoder-decoder structure to the input frustum points. Because GeoConv can only extract the intra-object features, with the increased

downsampling scale, PointNet (Qi et al., 2017) is used in our backbone to extract inter-object features. The semantic cues learned from 2D images are also leveraged for segmentation. Such information is encoded as a one-hot class vector and concatenated with the learned global features, and then back-propagated to point-wise features for the per-point class labeling. This segmentation network is a binary classification to segment background and foreground points.

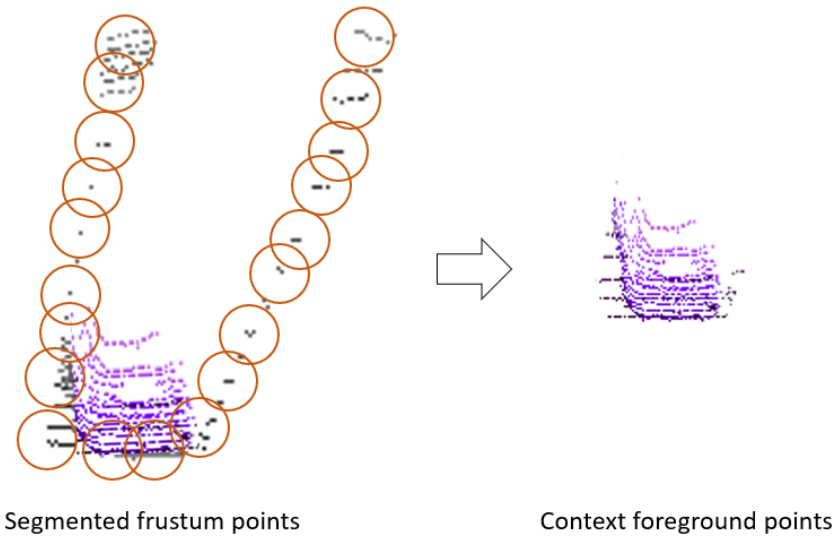

Figure 5.3: Context point collection.

As mentioned in PointRCNN (Shi et al., 2019), the context information around the object can improve the bounding box reasoning accuracy. Thus, to collect the context points from background points, we propose a context point collection method with an efficient and effective performance. For each background point, we collect its 16 nearest neighbours, as shown in Figure 5.3. If there has at least 1 foreground point, this background point is labeled as a context point. Query ball search (Qi et al., 2017) and KNN (Qi et al., 2017) are commonly used as neighbouring search methods. KNN searches the nearest neighbours without considering the distance. Thus, background points that are far away from the foreground point have a potential to be selected as the context points. These points have limited contributes to the object detection. To avoid such contamination, query ball search with 0.9m radius is experimentally selected as the neighbouring search method. This method not only selects

nearby background points and but also maintain the object geometric attributes. All the context points and foreground points are combined as the context foreground points. The experimental results demonstrate the effective of this method.

5.1.2 Residual Centre Estimation and Bounding Box Prediction

F-PointNet (Qi et al., 2018) has demonstrated the importance of coordinate transformations in enhancing the object detection performance. Those transformations can align the points in a set of constrained and canonical frames. Specifically, the object centre oriented transformation can help the 3D detectors better exploit the object geometric attributes, such as symmetry and planarity. Within the obtained context foreground points, we follow F-PointNet (Qi et al., 2018) to normalize these points to a local coordinate by subtracting their mean coordinates to boost the translational invariance. Then these points are input to a T-Net (Qi et al., 2017) to predict the residual box centre. The estimated residual centre can be derived as:

$$C_{pre} = C_{foreground} + C_{t-net} \tag{5.1}$$

where the $C_{foreground}$ represents the mean xyz of the foreground points. The context points are not considered in calculating the $C_{foreground}$ to make the predict centre closer to the object part. Then the normalized points are transformed into the predicted object centre for the bounding box prediction with canonical coordinates.

To predict an accurate bounding box, the bounding box prediction network should consider the local and global features of the object. The local feature encodes the object information, while the global feature provides the surrounding information of the object. Although we have added context points to the foreground, the information extracted from the context foreground points contains more information about the object. Features learned from the frustum points are more suitable to represent the global feature. Thus, the global features extracted from the foreground segmentation network are concatenated with the local global features extracted from the canonical context foreground points to predict the bounding box parameters. PointNet (Qi et al., 2017) is selected as the bounding box prediction network. In addition, the reflectance and semantic feature that learned from 2D proposals are also encoded for bounding box prediction. The experimental results demonstrate the effective of this network.

In this algorithm, a 3D bounding box is represented as $(x, y, z, h, w, l, \theta, score)$, where (x, y, z) is the object centre location, (h, w, l) is the object size (height, width, length), θ is the object orientation, and $score$ represents the objectness score. Following F-PointNet (Qi et al., 2018), we use a hybrid of classification and regression formulation. For angle prediction, we pre-define N_a and N_s as equally split angle and size bins and classify the proposal angle and size into different bins. Residual is regressed with respect to the bin value. N_a is set to 12 and N_s is set to 8 in our experiments. The bounding box prediction network outputs $3 + 4 \times N_s$ $+2 \times N_a + 2$.

5.1.3 Amodal Bounding Box Refinement

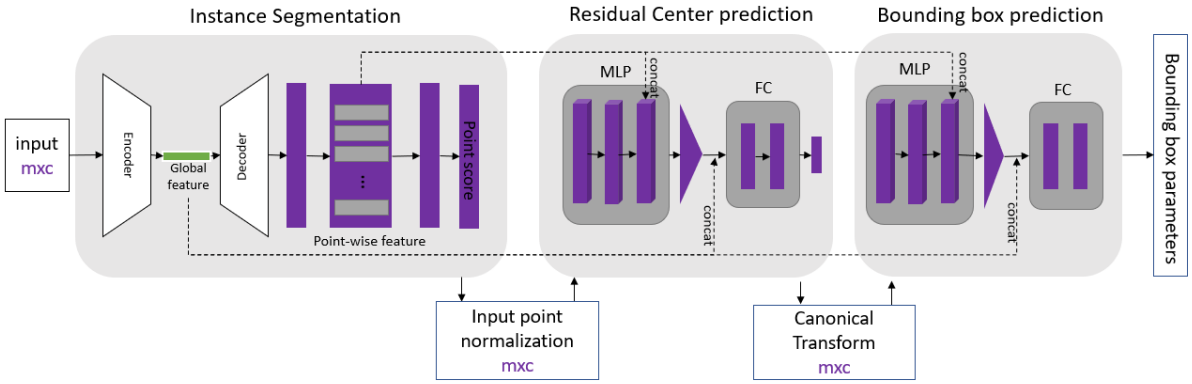

Figure 5.4: Refinement networks.

Although 2D region proposals detected by existing mature 2D detectors are precise enough, they cannot bound the object instance accurately. Larger 2D boxes contain the whole object instances but also include more background occlusions or clutters, while smaller 2D boxes contain less background noises but cannot provide the complete 3D object instances. To compensate this, in the refinement stage, we collect the points in the point clouds directly according to the estimated 3D boxes, instead of relying on the 2D boxes. The estimated bounding boxes are applied to recollect context foregrounds in point clouds only. Specifically, following Frustum ConvNet (Wang and Jia, 2019), we expand each estimated box by a specified factor -we set the factor as 1.2 in this work and normalize points inside the expanded box by translation and rotation.

To further improve the 3D detection performance, the point-wise, local and global features are considered to extract fine-grained box information. The refinement pipeline is similar to the detection pipeline in the first stage, as shown in Figure 5.4. The obtained points contain object and limited context information, we can see these points as context foreground points. Thus, the context point extraction module is removed in this stage. All the input points are utilized to learn the bounding box information. Point-wise features learned in the segmentation stage are concatenated with the MLP features learned in residual centre prediction network and bounding box prediction network for max-pooling. The global feature obtained in the segmentation stage is concatenated with the max-pooled feature in the bounding box prediction network for box parameter reasoning. The predicted box parameters are optimized with the same loss function in the detection stage.

5.1.4 Training with Multi-task Loss

Our point-based detection framework consists of two stage networks, one for bounding box prediction, the other for bounding box refinement. Each stage networks are optimized with a multi-task loss, which is composed of the segmentation loss, centre regression loss, bounding box loss (Qi et al., 2018), corner loss (Qi et al., 2018) and objectness loss. Both the objectness loss and the semantic segmentation loss are two-class cross-entropy loss. We adopt the similar bin-based classification and regression loss (Qi et al., 2018) for box optimization. The box loss is composed of the centre regression, heading estimation and size estimation sub-losses using Huber loss:

$$L_{box} = L_{center-reg} + L_{ang-cls} + L_{angle-reg} + L_{size-cls} + L_{size-reg} + L_{corner} \qquad (5.2)$$

where $L_{center-reg}$ is the residual central loss for the predicted centre, $L_{ang-cls}$ and $L_{angle-reg}$ represent the loss of classification and regression for the predicted angle, respectively. $L_{size-cls}$ and $L_{size-reg}$ represent the loss of classification and regression for the predicted size, respectively. The corner loss L_{corner} is derived from the distance between the predicted corners $Corner_{pre}$ and the groundtruth corners $Corner_{gt}$ (Qi et al., 2018):

$$L_{corner} = \sum_{k=1}^{8} \left\| Corner_{pre} - Corner_{gt} \right\| \tag{5.3}$$

The groundtruth of the objectness score is defined using the distance of the predicted box centre to the groundtruth box centre. If the distance is larger than 0.3m, the groundtruth label is set to 1, otherwise, the label is set to 0. Within such operation, the predicted score has geometric correlation with the predicted bounding box. Thus, the total loss L_{total} for each network is as follows:

$$L_{total} = L_{seg} + L_{center-reg} + L_{box} + L_{objectness} + L_{corner} \tag{5.4}$$

where L_{seg} is the segmentation loss, $L_{center-reg}$ is the regressions loss for predicted centre in residual centre prediction module, $L_{objectness}$ is the objectness loss.

5.2 Experiments

In this part, the experimental performance of our proposed method is presented and analyzed. In Section 5.2, we introduce the experimental setting of our approach, including datasets, evaluation criteria, and implementation details. Then, the object detection results are presented in Section 5.2.1. Ablation studies to analyze the proposed modules are conducted in Section 5.2.2. Optimizer, memory usage and timing are provided in Section 5.2.3. Finally, discussions about the merits and demerits of our method are presented in Section 5.2.4.

5.2.1 Experimental Setting

Datasets. The performance of our method is evaluated on two datasets: KITTI (Geiger et al., 2013) and SUN-RGBD (Song et al., 2015). The KITTI dataset was collected in outdoor scenes by a moving platform equipped with cameras, laser scanners, GPS and IMU. Thus, it can provide LiDAR points and corresponding images with high accuracy. This dataset contains 7481/7518 training/testing samples. In this paper, we follow F-PointNet (Qi et al., 2018) to split the training samples into train split (3712 samples) and val split (3769 samples). Detection results on val split are reported and compared with other state-of-art methods.

SUN-RGBD is collected using multiple different RGB-D cameras with varying resolutions from different indoor scenes. It contains 5,285 training images and 5,050 testing images, respectively. There are 37 object categories labeled with amodal oriented 3D bounding boxes. We report model performance on the testing set. Point cloud data are acquired following the method provided by F-PointNet (Qi et al., 2018). Detection results on the 10 most common categories are reported.

Evaluation Criteria. Following F-PointNet (Qi et al., 2018), the average precision metric AP_{3D} of 3D detection results is adopted as our evaluation criteria. The predicted bounding box B_p is treated as a valid detection result only its 3D overlap area (IoU) between the predicted bounding box B_p and ground truth bounding box B_{gt} exceeds a certain ratio. IoU is calculated using the following evaluation metric:

$$IoU = \frac{Area(B_p \cap B_{gt})}{Area(B_p \cup B_{gt})} \tag{5.5}$$

Predicted bounding boxes with 3D IoU results exceeding 0.7 are used to evaluate the car detection performance in KITTI dataset and the IoU threshold for all classes in SUN-RGBD dataset is 0.25.

5.2.2 Implementation Details

We use the 2D detection results of KITTI validation provided by F-PointNet (Qi et al., 2018) and the 2D detection results of SUN-RGBD validation provided by Frustum-ConvNet (Wang and Jia, 2019) to extract frustum points. Data augmentation is applied to the detected 2D bounding boxes by translation and scaling during training. The input to the first stage detection networks is 2,048 points for KITTI. The number of inputs in the second stage refinement networks for KITTI is 512. The input to the SUN-RGBD framework is 2048 points. Similar random flipping and shifting (Qi et al., 2018) are adopted to these points.

The details of the detection and refinement networks are shown in Figure 5.5. The point cloud segmentation network is a multi-scale encoder-decoder structure (Qi et al., 2017). The encoder is constructed with set abstract (SA) module, while the decoder is composed with feature propagation (FP) module. As for the detection networks, when the sampling radius is

set to 0.2m, the GeoConv (Li et al., 2020) is used to learn intra-object features. For other sampling radius, the PointNet is used to map inter-object features. But for the refinement networks, only context foreground points with different geometric distributions are input for point segmentation. To improve the refinement performance, point-wise features learned in segmentation network are concatenated with the object global feature for bounding box prediction. Such features are sensitive to the incomplete geometric shape variances. Because GeoConv cannot extract the expressive intra-object features, the PointNet (Qi et al., 2017) is used in all multi-scale set abstract layers to learn local and global features in the stage-2 segmentation network.

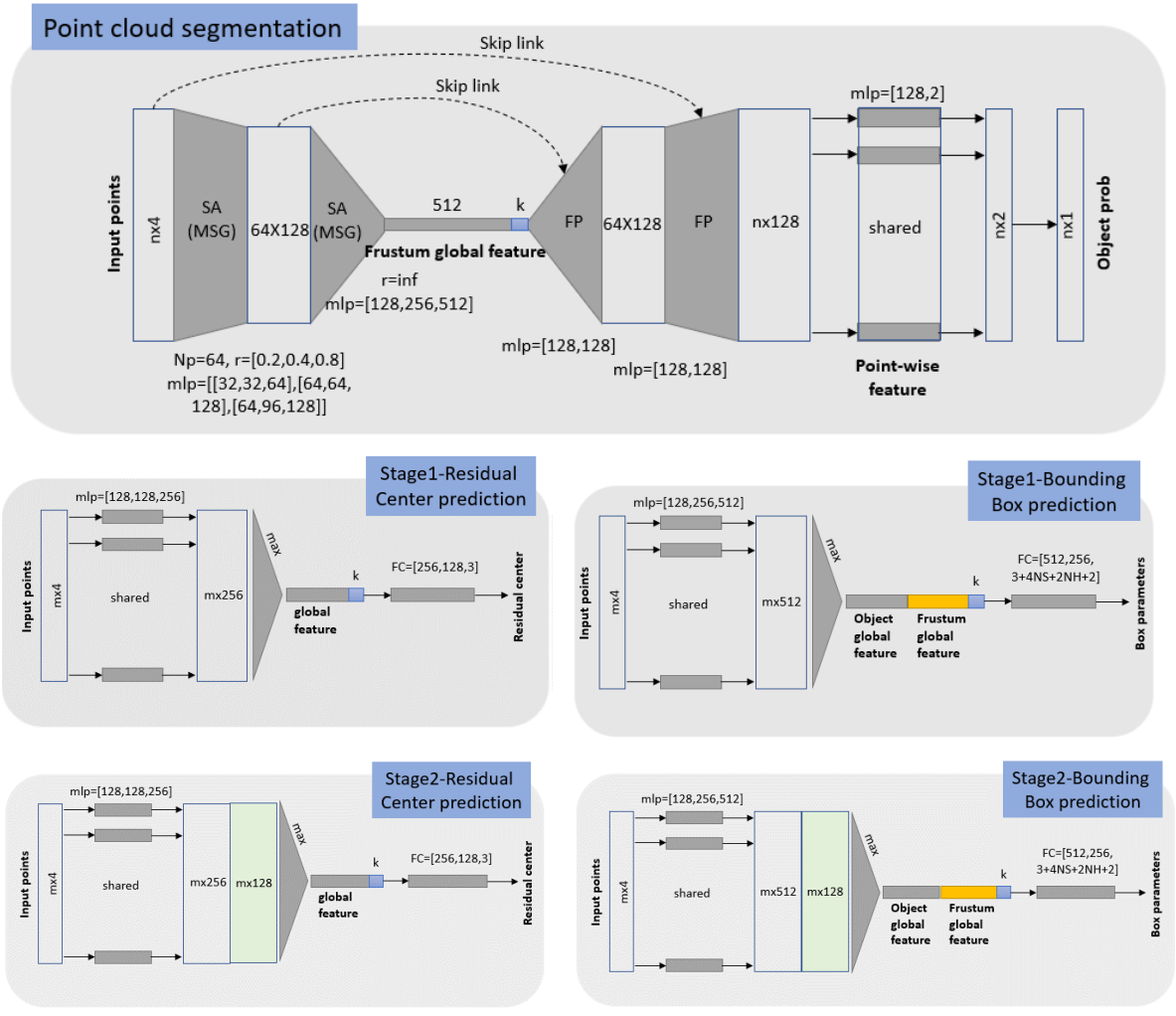

Figure 5.5: The details of the detection and refinement networks.

Residual centre prediction and bounding box prediction networks in both two stages are constructed based on PointNet (Qi et al., 2017). The learned object global features in bounding box prediction networks of detection and refinement networks are all concatenated with the frustum global feature and one-hot 2D semantic cues for bounding box parameter prediction. For the refinement networks, the point-wise features learned in segmentation stage are concatenated with the per-point features learned from the normalized points. These features are maxpooled to generate global features for residual centre prediction and bounding box parameter prediction.

5.2.3 Object Detection Results

(1) KITTI Detection Results

Table 5.1: 3D detection, 3D localization, and 2D detection results.

	Input	3D (%)			BEV (%)			2D (%)		
		Easy	Mode-rate	Hard	Easy	Mode-rate	Hard	Easy	Mode-rate	Hard
VoxelNet	LiDAR	82.0	65.5	62.9	89.6	84.8	78.6	-	-	-
IPOD	LiDAR	84.1	76.4	75.3	88.3	86.4	84.6	-	-	-
PointRCNN	LiDAR	88.9	78.6	**77.4**	-	-	-	-	-	-
MV3D	LiDAR +RGB	71.3	62.7	56.6	86.6	78.1	76.7	-	-	-
ContFusion	LiDAR +RGB	86.3	73.3	67.8	95.4	87.3	82.4	-	-	-
F-PointNet	LiDAR +RGB	83.8	70.9	63.7	88.2	84.0	76.4	96.5	90.3	87.6
Frustrum ConvNet	LiDAR +RGB	**89.0**	**78.8**	77.1	90.2	**88.8**	**86.8**	96.5	90.3	87.6
Ours	LiDAR +RGB	88.8	78.1	75.3	**90.3**	88.6	79.9	**98.1**	**90.4**	**87.9**

We evaluate our detection framework on the KITTI val split with 3769 samples, and the experimental results are shown in Table 5.1. Existing methods with LiDAR and RGB inputs

such as F-PointNet (Qi et al., 2018), MV3D (Chen et al., 2017), ContFusion (Liang et al., 2018), Frustrum ConvNet (Wang and Jia, 2019) and LiDAR only inputs such as VoxelNet (Zhou and Tuzel, 2018), IPOD (Yang et al., 2018), and PointRCNN (Shi et al., 2019) are compared to demonstrate the effectiveness of our proposed method. Compared with these methods, our method achieves the compatible results as Frustrum ConvNet (Wang and Jia, 2019) and PointRCNN (Shi et al., 2019) and better results than the remaining approaches in easy and moderate difficulties in 3D object detection and localization tasks. Although we use the same detector as F-PointNet (Qi et al., 2018) and Frustrum ConvNet (Wang and Jia, 2019), our 2D detection results improved about 1.4% AP in easy difficulty. The main difference is that we use the sum of 2D proposal score and 3D detection score as the final objectness score to post-process the detected results. Boxes with higher 2D and 3D objectness score have higher probability to be detected.

(2) SUN-RGBD Detection Results

We compare our detection results with existing state-of-art algorithms on SUN-RGBD dataset, such as DSS (Song and Xiao, 2016), COG (Ren and Sudderth, 2016), 2Ddriven3D (Lahoud and Ghanem, 2017), PointFusion (Xu et al., 2018), F-PointNet (Qi et al., 2018), 3D-Latent (Ren et al., 2018), VoteNet (Qi et al., 2019), Frustum ConvNet (Wang and Jia, 2019), GRNet (Li et al., 2020), etc. Compared with outdoor LiDAR points, indoor points lifted from the RGB-D camera are denser and indoor objects commonly exist together and have similar shapes. Thus, we only use the refinement networks to detect 3D objects from SUN-RGBD frustum points. Context extraction method is not applied in SUN-RGBD detection framework. As shown in Table 5.2, our proposed method achieves the best result with mAP 58.5%. Such improvements mainly come from the discriminative point-wise, local and global feature exploration for amodal bounding box prediction.

Table 5.2: 3D object detection AP (%) on SUN-RGBD val set.

Methods	Bathtub	Bed	Bkshf	Chair	Desk	Dresser	Nitstd	Sofa	Table	Toilet	mAP
DSS	44.2	78.8	11.9	61.2	20.5	6.4	15.4	53.5	50.3	78.9	42.1
COG	58.3	63.7	31.8	62.2	45.2	15.5	27.4	51.0	51.3	70.1	47.6
2Ddriven3D	43.5	64.5	31.4	48.3	27.9	25.9	41.9	50.4	37.0	80.4	45.1
PointFusion	37.3	68.6	37.7	55.1	17.2	24.0	32.3	53.8	31.0	83.80	45.4
3D-Latent	76.2	73.2	32.9	60.5	34.5	13.5	30.4	60.4	55.4	73.7	51.0
F-PointNet	43.3	81.1	33.3	64.2	24.7	32.0	58.1	61.1	51.1	90.9	54.9
VoteNet	74.4	83.0	28.8	75.3	22.0	29.8	62.2	64.0	47.3	90.1	57.7
Frustum ConvNet	61.3	83.2	36.5	64.4	29.7	35.1	58.4	66.6	53.3	87.0	57.6
GRNet	76.8	84.3	29.3	76.2	26.0	26.1	59.2	64.8	51.1	90.4	58.4
Ours	59.6	82.7	36.3	66.6	32.8	33.6	59.6	67.6	55.2	90.8	**58.5**

5.2.4 Ablation Studies

To demonstrate the effectiveness and importance of the context foreground extraction method, feature fusion in detection networks and the effectiveness of refinement networks, some ablation studies are conducted. When testing each module, the remaining modules remain unchanged. The followings are the detailed evaluation of these modules.

(1) Context foreground point extraction

The context information for each object is critical for 3D learners to differentiate the target from background clutters. Thus, in this paper, we propose a context point extraction method that can attribute to the 3D object detection performance in detection networks. In our method, the extraction radius is an important hyperparameter. In achieved with a 0.9 searching radius. Especially in the easy and moderate difficulties, our method has 2.2% and 1% improvements than the performances without context extraction. However, the improvement for hard difficulty is limited, with less than 1% improvement. The main reason is that the hard difficulty has sparse and limited points, which are gathered closer to the centre. Such radius cannot collect enough context points for these objects.

Table 5.3, we test different radius to show their differences. Within the context extraction method, the best performance was achieved with a 0.9 searching radius. Especially in the easy and moderate difficulties, our method has 2.2% and 1% improvements than the performances without context extraction. However, the improvement for hard difficulty is limited, with less than 1% improvement. The main reason is that the hard difficulty has sparse and limited points, which are gathered closer to the centre. Such radius cannot collect enough context points for these objects.

Table 5.3: Ablation studies of the context extraction method.

Context radius	3D (%)			BEV (%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
No context	85.7	75.5	67.2	89.5	87.5	78.8
0.8m	86.3	75.9	67.5	89.2	87.2	78.7
0.9m	**88.0**	**76.5**	**67.9**	**90.1**	**87.9**	**79.1**
1m	87.9	76.4	67.8	90.0	87.5	78.8
1.1m	87.6	76.2	67.6	90.0	87.6	79.0

(2) Feature fusion

The local and global features of the object are crucial to predict an accurate amodal bounding box. The local feature contains the object information, while the global feature encodes the surrounding information of the object. In F-PointNet (Qi et al., 2018), the global feature from frustum points is not considered in bounding box prediction. In this paper, we use the global feature learned from the point cloud segmentation network to represent the frustum point global feature. This feature is concatenated with the object global feature and one-hot semantic cues from the 2D proposal for amodal bounding box prediction in detection networks. The results in Table 5.4 demonstrate that such fusion can improve around 1.5% for easy and moderate difficulties. For hard difficulty, the global feature contributes limited. More specific global feature with small range around the hard difficulty may largely improve the detection performance.

Table 5.4: Ablation studies of the effects of feature fusion.

	3D (%)			BEV (%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
Local & semantic cues	86.7	75.5	67.7	89.5	86.2	78.5
Local & global & semantic cues	**88.0**	**76.5**	**67.9**	**90.1**	**87.9**	**79.1**

(3) Refinement networks

As mentioned in the context point extraction and feature fusion ablation experiment analysis, these two approaches for hard difficulty improvement is limited. To provide more object-specific context foreground points and compensate the incorrect 2D proposals, the refinement networks are proposed to learn more specific object and context information for bounding box refinement. As shown in Table 5.5, the detection performance of hard difficulty improves about 7%, while easy and moderate increase around 0.8% and 1.6%, respectively. The improvement for localization accuracy is limited, within around 0.8% increases.

Table 5.5: Ablation studies of the effectiveness of refinement networks.

	3D (%)			BEV (%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
Ours-detection	88.0	76.5	67.9	90.1	87.9	79.1
Ours-detection +refinement	**88.8**	**78.1**	**75.3**	**90.3**	**88.6**	**79.9**

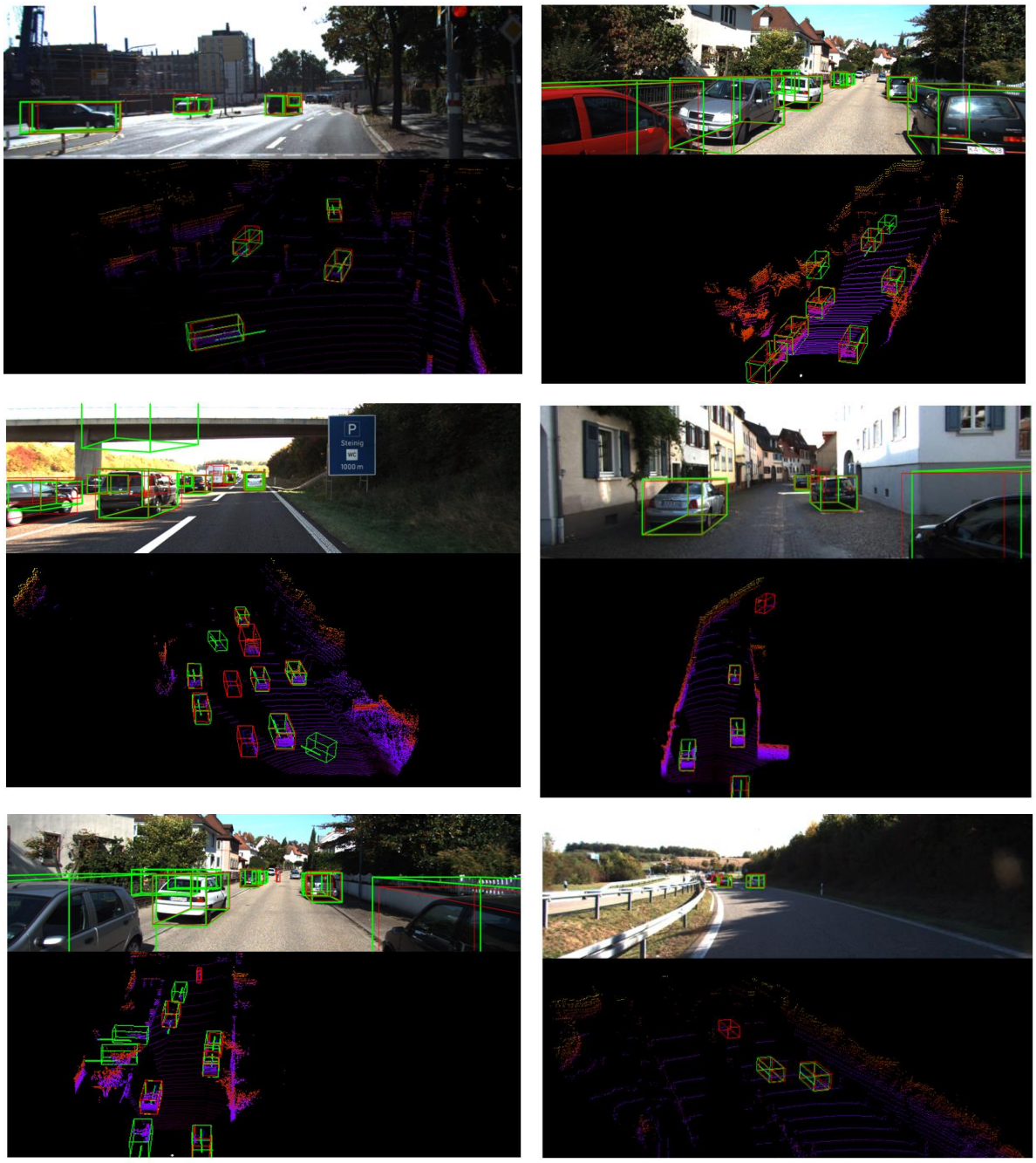

Figure 5.6: Visualization of our results on KITTI val set.

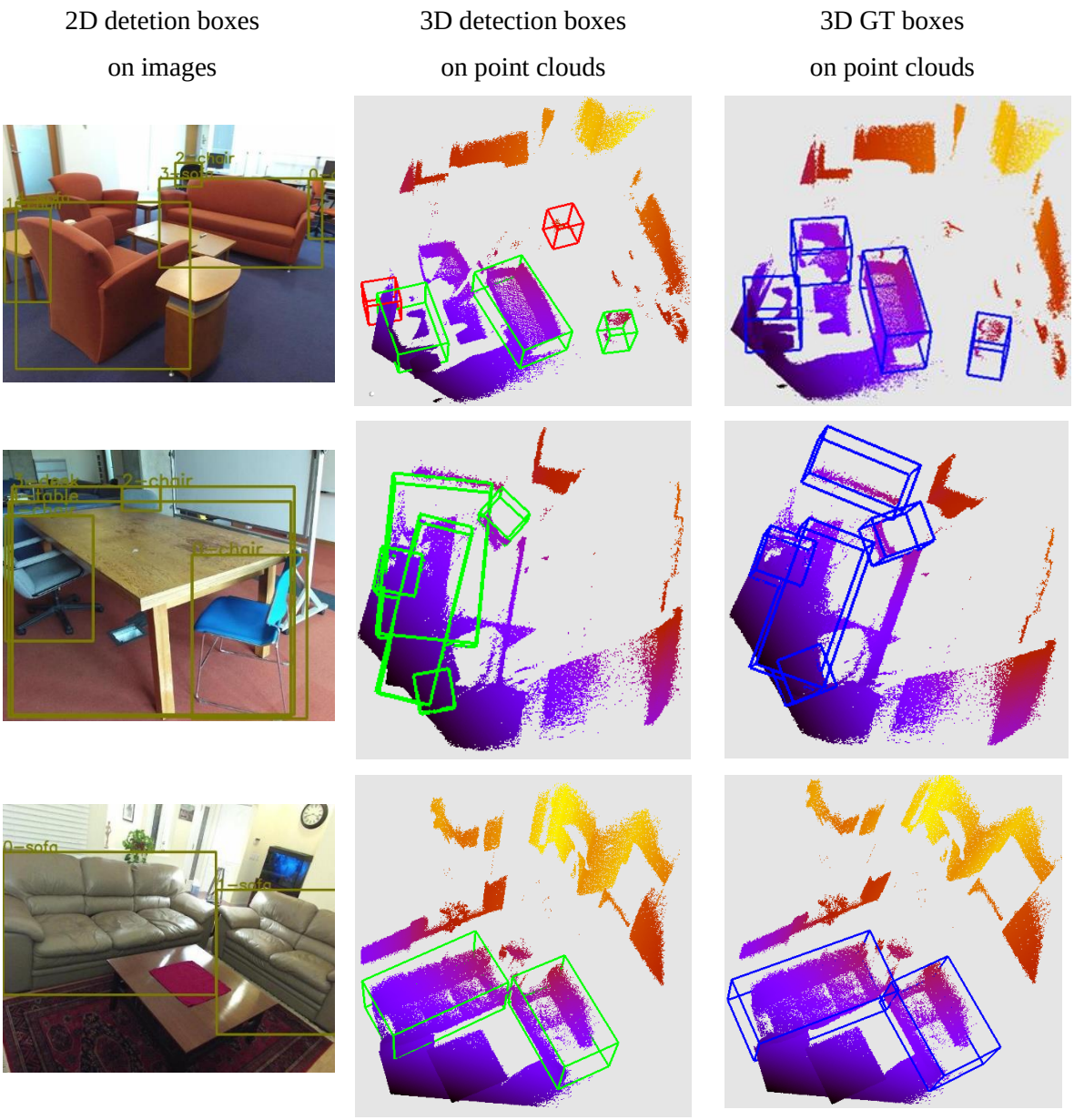

Figure 5.7: Visualization of our results on SUN-RGBD val set.

5.2.5 Optimizer, Timing and Hyperparameter Setting

We implement our detection and refinement networks with Python 3.5 and TensorFlow 1.8 on one GTX 1080ti GPU. ADAM optimizer (Kingma and Ba, 2014), with an initial learning rate of 0.003 and a 0.1 decay rate, is adopted in KITTI dataset. Table 5.6 shows the comparison

results of our method and F-PointNet (Qi et al., 2018) and Frustum ConvNet (Wang and Jia, 2019) on timing and hyperparameter setting. The batch size was set to 24 for training and testing with 100 epochs for two stage networks respectively. Although we have detection and refinement networks, the total training time only consumes around 14 hours totally to get the desired detection results.

Table 5.6: Timing and hyperparameter setting on KITTI dataset.

	#GPU	Training Time	Input points	# Epoch	Learning rate
F-PointNet	1	~3 days	2048	200	0.001
Frustum ConvNet	1	~1 day	2048	50	0.001
Detection network (ours)	1	~10 hours	2048	100	0.003
Refinement network (ours)	1	~4 hours	512	40	0.003

5.3 Discussion

In Figure 5.6 and Figure 5.7, some representative outputs of our method in KITTI and SUN-RGBD datasets are presented. In most cases, the 3D box can be accurately detected. Even for partial data or overlapping 2D objects, as shown in image data, our model can predict the amodal bounding box precisely. Even for some very partial examples which hard for 2D detectors to predict, our method can localize their 3D boxes with remarkable accuracy as long they have 2D proposals.

However, there are several failure examples, which need further improvements. The first mistake comes from the misdetection in 2D images. If the positive object is not detected in the 2D image, its 3D bounding box cannot be predicted via our point-based detection networks. The second failure is the inaccurate pose and size estimation for hard difficulty with limited object points. Due to the lack of enough object information in this case, the orientation and pose are hard to be optimized.

In our paper, we have tested the indoor and outdoor point cloud dataset for object detection. The difference between these two datasets corresponds to the two different detection

frameworks. The KITTI points are sparse and large-scale, the extracted frustum points contain more background information. Thus, the context extraction module and refinement networks can greatly improve the detection performance, especially for the hard difficulty. But in SUN-RGBD dataset, point clouds lifted from RGB-D data are dense and small-scale. The extracted frustum points are more similar to the enlarged predicted bounding box in the KITTI detection model. As a result, using the refinement networks only on SUN-RGBD frustum points can achieve remarkable performance.

5.4 Chapter Summary

In this chapter, a 2D-driven frustum-based two-stage object detection framework to detect objects in indoor and outdoor environments was proposed. A 2D proposal was used to extract a frustrum 3D space, points in such space were leveraged via detection networks to estimate a coarse amodal bounding box. To compensate the inaccurate 2D proposals, refinement networks were followed to refine the estimated bounding box. Both semantic features from 2D images and the object and context information in 3D space were explicitly exploited to enhance the 3D detection performance. We have validated our model on the KITTI val set with 88.8%, 78.1%, and 75.3 % 3D AP for easy, moderate, and hard difficulties, respectively. On SUN-RGBD dataset, our algorithm achieves the leading performance with 58.5% mAP. Ablation studies were provided to demonstrate the effectiveness of each designed module. However, the detection performance is constrained by the 2D detection accuracy. In future, proposals in 2D and 3D data will be extracted parallelly to overcome the misdetection and inaccurate detection in 2D images.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

The wide application of 3D sensing in automatic perception, urban modeling, and infrastructure survey, and the rapid development of deep learning in robust and discriminate feature extraction have prompted the emerging of 3D information extraction using deep learning techniques. 3D data are commonly represented as point clouds, which are massive and irregular. This poses a great challenge for applying deep learning to process these data. Besides, the 3D data collected by different sensors in different scenes show multiple variations, e.g., point density, point distribution, and geometric shape. While 3D data is often in the form of point clouds, how to represent point clouds and which deep model to use for 3D information extraction remains an open problem.

This dissertation provides a set of deep learning frameworks for 3D information extraction, specific for point cloud segmentation and object detection tasks. 2D-driven 3D object detection is also explored to demonstrate the effectiveness of 2D image leveraging to assist 3D object detection from point clouds. Data collected by LiDAR and RGB-D sensors in multiple indoor and outdoor scenes are studied to validate the accuracy and efficiency of our proposed algorithms.

For point cloud segmentation, an end-to-end geometric graph convolution architecture is constructed on the graph representation of point clouds to predict the per-point semantic label. It employs a multiscale hierarchical architecture by operating TGConv on neighbours at multiple scales and a CRF layer combined within the output layer to further improve the segmentation result. Qualitative and quantitative experimental results on the ScanNet and S3DIS indoor datasets and the Paris-Lille-3D outdoor benchmark demonstrate the effectiveness of the proposed method.

For object detection, an end-to-end point cloud geometric relation framework, focusing on 3D object detection, is proposed. The geometric feature among object points and the relation feature between different objects are explored to enhance the detection performance. A centralization module with a scalable loss function and an object-relation learning module are applied to predict bounding box parameters. Experimental results demonstrate the effectiveness of the proposed algorithm on SUN-RGBD and ScanNetV2 indoor datasets.

To leverage 3D images for 3D object detection, a 2D-driven frustum-based two-stage object detection architecture is presented to detect objects in indoor and outdoor environments. A 2D proposal is used to extract a frustrum 3D space with the known camera projection matrix. Points in this space are leveraged via detection networks to estimate a coarse amodal bounding box. Then the refinement networks are followed to refine the estimated bounding box. Both semantic features from 2D images and the object and context information in 3D space are fused to boost the 3D detection performance. Experimental results on the KITTI val set and SUN-RGBD datasets show the capability of the proposed method.

6.2 Contributions

This dissertation has made several contributions for deep learning based 3D information extraction in point cloud segmentation and object detection tasks.

For point cloud segmentation, an end-to-end geometric graph convolution architecture (TGNet) for per-point semantic labeling has been presented. There are four main contributions:

- A novel convolutional filter that can capture local correlations described by neighbourhood features and local geometric features is proposed. These features can enhance the filter's shape description capability.
- Point features are extracted in a hierarchically multiscale way, which can ensure the information from different scales can be combined together to increase the segmentation performance.
- A CRF layer is added after the output layer when constructing the end-to-end trainable framework.

- The TGNet achieved state-of-the-art results in three cases with 62.2% average accuracy on ScanNet, 57.8% and 68.2% mIoU on S3DIS and Paris-Lille-3D datasets, respectively.

For object detection, an end-to-end point cloud geometric relation framework (GRNet) focusing on 3D object detection has been proposed. The main contributions of GRNet are as follows:

- A novel geometric convolution is proposed and applied in a bottom-up backbone network. Intra-object geometric features and inter-object relation features for each representative point are extracted in a hierarchical way.
- A centralization module is presented to centralize object surface points to its centre. This contributes to an improved bounding box prediction.
- An object relation learning module is introduced to exploit the relation feature between proposals for better bounding box reasoning.
- The GRNet achieved state-of-the-art 3D detection results with 59.1% mAP@0.25 and 39.1% mAP@0.5 on ScanNetV2 dataset, 58.4% mAP@0.25 and 34.9% mAP@0.5 on SUN RGB-D dataset.

To leverage 2D images for 3D object detection, a 2D-driven frustum-based two-stage 3D object detection architecture has been constructed. This framework is featured with the following four contributions:

- To leverage the 2D images of the point clouds, each 2D proposal is lifted to a frustrum 3D space and points in such space are collected. Then, these frustum points are used to estimate the coarse bounding boxes of 3D objects.
- To compensate the inaccurate 2D proposals, the refinement networks are proposed to refine the estimated bounding box.
- Both semantic features from 2D images and the context information in 3D space are fused to enhance the 3D detection performance.

- This framework achieved compatible results on the KITTI val set with 88.8%, 78.1%, and 75.3 % 3D AP for easy, moderate, and hard difficulties, respectively. On SUN-RGBD dataset, this algorithm achieved the leading performance with 58.5% mAP.

6.3 Discussions and Recommendations for Future Studies

This thesis has proposed three novel deep learning frameworks for point cloud segmentation and object detection tasks, which can effectively extract 3D information from indoor and outdoor scenes. However, there still exists a huge gap between cutting-edge results and human-level performance. Although there is much work to be done, we mainly summarize the remaining challenges specific for data, deep architectures, and tasks, and then discuss the corresponding future researches in the following six aspects:

Robust Data Representation: Although there are several effective data representations such as voxels (Maturana and Scherer, 2015), point clouds (Qi et al., 2017; Qi et al., 2017) , graphs (Xu et al., 2018; Wang et al., 2019), 2D views (Kanezaki et al., 2018), or novel 3D data representations (Le and Duan, 2018; Li et al., 2019; Mescheder et al., 2019), there has not yet agreed on a robust and memory-efficient 3D data representation. As for point clouds and graphs which have been explored in this thesis, the permutation invariance and the computation capability limit the processable quantity of points, which inevitably constrains the segmentation and detection accuracy and efficiency of the proposed deep models. Thus, deep representation learning focuses on improving existing 3D data representations (Mescheder et al., 2019) or proposing novel 3D data representations (He et al., 2019; Mescheder et al., 2019) based on exploiting the intrinsic and geodesic structure of data in local 3D space remains an interesting and challenging task.

Multi-source Data Fusion: To compensate the absence of semantic, textual and incomplete information in 3D points, point clouds are fused with 2D images for 3D object detection in this thesis. Besides, there also exists a fusion between data acquired by low-end LiDAR (e.g., Velodyne HD-16E) and high-end LiDAR (e.g., Velodyne HD-64E) sensors. However, there exist several challenges in fusing these data: the first is that the sparsity of point clouds causes the inconsistent and missing data when fusing multi-source data; the

second is that the proposed data fusion scheme using deep learning knowledge is processed in a separate line, which is not an end-to-end scheme. Thus, how to fuse multi-source data indicates a valuable research direction.

Effective and More Efficient Deep Frameworks: Due to the limitation of memory and computation facilities, effective and efficient deep learning architectures are crucial for the wide applications in automatic sensing and localization. Although the proposed models have achieved several significant accuracy and efficiency improvements, such as TGNet and GRNet, the real-time segmentation and detection tasks are not achieved. Lightweight and compact architecture designing should be considered to reduce the computation cost of these proposed models.

Context Knowledge Extraction: Due to the sparsity of point clouds and incompleteness of scanned objects, detailed context information for objects is not fully exploited. For example, the semantic context of vehicles is crucial for autonomous navigation, but the proposed 2D-driven 3D object detection method cannot extract such information completely from point clouds. Besides, the proposed framework cannot solve the sparsity and incompleteness problems for context information extraction in an end-to-end trainable way.

Multi-task Learning: The approaches related to 3D information extraction can be classified into several tasks, such as scene segmentation, object detection (e.g., cars, pedestrians, traffic lights, etc.) and classification (e.g., road markings, traffic signs). All these results are commonly fused together to report a comprehensive result in product and model generation (Janai et al., 2017). However, the proposed three models cannot combine these multiple point cloud tasks together. Thus, the inherent information among them is not fully exploited and used to generalize better models with less computation.

Weakly Supervised/Unsupervised Learning: The three proposed models are constructed under supervised modes using labeled data for per-point labeling or 3D bounding box prediction. However, there are some limitations of these fully supervised models. The first is the limited availability of high quality, large scale, and enormous general objects datasets and benchmarks. The second is the fully supervised model generalization capability which is not

robust to unseen or untrained objects. Weakly supervised (Yew and Lee, 2018) or unsupervised learning (Sauder and Sievers, 2019; Shoef et al., 2019) should be developed to increase the model's generalization and solve the data absence problem.

Copyright Permissions

IEEE, as the publisher of the two manuscripts fully or partly adopted in Chapter 1, Chapter 2, Chapter 3, Chapter 5, and Chapter 6 allow the reuse of published papers in the thesis without formal permissions. Thus, the waivers of copyright from IEEE are achieved by the following statement:

Policy Regarding Thesis/Dissertation Reuse, from IEEE Copyright Clearance Centre

"The IEEE does not require individuals working on a thesis to obtain a formal reuse license; however, you may print out this statement to be used as a permission grant".

Elsevier, as the publisher of the two manuscripts fully or partly adopted in Chapter 2 and Chapter 4 allow the reuse of published papers in the thesis without formal permissions. Thus, the waivers of copyright from Elsevier are achieved by the following statement:

Policy Regarding Thesis/Dissertation Reuse in Elsevier Copyright:

"Authors can use their articles, in full or in part, for a wide range of scholarly, non-commercial purposes in a thesis or dissertation (provided that this is not to be published commercially)."

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J. and Devin, M., 2015. TensorFlow: Large-scale machine learning on heterogeneous distributed systems, *arXiv: 1603.04467*.

Ahmed, E., Saint, A., Shabayek, A. E. R., Cherenkova, K., Das, R., Gusev, G., Aouada, D. and Ottersten, B., 2018. Deep learning advances on different 3D data representations: a survey, *arXiv:1808.01462*, vol. 1.

Armeni, I. and Zamir, A. R., 2016. 3D semantic parsing of large-scale indoor spaces, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1534-1543.

Arnold, E., Aljarrah, O. Y., Dianati, M., Fallah, S., Oxtoby, D. and Mouzakitis, A., 2019. A survey on 3D object detection methods for autonomous driving applications, *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782-3795.

Beltran, J., Guindel, C., Moreno, F. M., Cruzado, D., Garcia, F. and La Escalera, A. D., 2018. BirdNet: A 3D object detection framework from LiDAR information, *Proceedings of the International Conference on Intelligent Transportation Systems*, pp. 3517-3523.

Benshabat, Y., Avraham, T., Lindenbaum, M. and Fischer, A., 2018. Graph based over-segmentation methods for 3D point clouds, *Computer Vision and Image Understanding*, vol. 174, pp. 12-23.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A. and Vandergheynst, P., 2017. Geometric deep learning going beyond Euclidean data, *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18-42.

Bruna, J., Zaremba, W., Szlam, A. and Lecun, Y., 2013. Spectral networks and locally connected networks on graphs, *arXiv:1312.6203v3* .

Chang, A. X., Dai, A., Funkhouser, T., Halber, M., Niebner, M., Savva, M., Song, S., Zeng, A. and Zhang, Y., 2017. Matterport3D: Learning from RGB-D data in indoor environments, *Proceedings of International Conference on 3D Vision*, pp. 667-676.

Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S. and Urtasun, R., 2016. Monocular 3D object detection for autonomous driving, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2147-2156.

Chen, X., Ma, H., Wan, J., Li, B. and Xia, T., 2017. Multi-view 3D object detection network for autonomous driving, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1907-1915.

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T. and Niener, M., 2017. ScanNet: Richly-annotated 3D reconstructions of indoor scenes, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2432-2443.

Dai, A. and Niesner, M., 2018. 3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation, *Proceedings of the European Conference on Computer Vision*, pp. 458-474.

Dai, A., Ritchie, D., Bokeloh, M., Reed, S., Sturm, J. and Niebner, M., 2018. ScanComplete: Large-scale scene completion and semantic segmentation for 3D scans, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4578-4587.

De Brabandere, B., Neven, D. and Van Gool, L., 2017. Semantic instance segmentation for autonomous driving, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 478-480.

Deng, H., Birdal, T. and Ilic, S., 2018. PPFNet: Global context aware local features for robust 3D point matching, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 195-205.

Deng, Z. and Latecki, L. J., 2017. Amodal detection of 3D objects: Inferring 3D bounding boxes from 2D ones in RGB-Depth images, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 398-406.

Dissanayake, M. W. M. G., Newman, P., Clark, S., Durrantwhyte, H. and Csorba, M., 2001. A solution to the simultaneous localization and map building (SLAM) problem, *Proceedings of the International Conference on Robotics and Automation*, vol. 17, pp. 229-241.

Dong, Z., Yang, B., Liang, F., Huang, R. and Scherer, S., 2018. Hierarchical registration of unordered TLS point clouds based on binary shape context descriptor, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 144, pp. 61-79.

Engelcke, M., Rao, D., Wang, D. Z., Tong, C. H. and Posner, I., 2017. Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks, *Proceedings of the International Conference on Robotics and Automation*, pp. 1355-1361.

Engelmann, F., Kontogianni, T., Schult, J. and Leibe, B., 2018. Know what your neighbors do: 3D semantic segmentation of point clouds, *Proceedings of the European Conference on Computer Vision*, pp. 395-409.

Everingham, M., Eslami, S. M., Van Gool, L., Williams, C. K. I., Winn, J. and Zisserman, A., 2015. The pascal visual object classes challenge: A retrospective, *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98-136.

Fan, H., Su, H. and Guibas, L. J., 2017. A point set generation network for 3D object reconstruction from a single image, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2463-2471.

Fey, M., Lenssen, J. E., Weichert, F. and Muller, H., 2018. SplineCNN: Fast geometric deep learning with continuous B-Spline kernels, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 869-877.

Fujiwara, K., Sato, I., Ambai, M., Yoshida, Y. and Sakakura, Y., 2018. Canonical and compact point cloud representation for shape classification, *arXiv:1809.04820v1*.

Gao, H., Cheng, B., Wang, J., Li, K., Zhao, J. and Li, D., 2018. Object classification using CNN-based fusion of vision and LiDAR in autonomous vehicle environment, *IEEE Transactions on Industrial Informatics*, vol. 14, no. 9, pp. 4224-4231.

Geiger, A., Lenz, P., Stiller, C. and Urtasun, R., 2013. Vision meets robotics: The KITTI dataset, *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231-1237.

Guan, H., Li, J., Cao, S. and Yu, Y., 2016. Use of mobile LiDAR in road information inventory: A review, *International Journal of Image and Data Fusion*, vol. 7, no. 3, pp. 219-242.

Guan, P. and Neumann, U., 2016. 3D point cloud object detection with multi-view convolutional neural network, *Proceedings of the International Conference on Pattern Recognition*, pp. 585-590.

Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu and M. Bennamoun, 2020. Deep learning for 3D point clouds: a survey, I*EEE Transactions on Pattern Analysis and Machine Intelligence*, doi: 10.1109/TPAMI.2020.3005434.

Gupta, S., Girshick, R., Arbeláez, P. and Malik, J., 2014. Learning rich features from RGB-D images for object detection and segmentation, *Proceedings of the European Conference on Computer Vision*, pp. 345-360.

Hackel, T., Wegner, J. D. and Schindler, K., 2017. Joint classification and contour extraction of large 3D point clouds, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 130, pp. 231-245.

He, T., Huang, H., Yi, L., Zhou, Y., Wu, C., Wang, J. and Soatto, S., 2019. GeoNet: Deep geodesic networks for point cloud analysis, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6888-6897.

Hornik, K., 1991. Approximation capabilities of multilayer feedforward networks, *Neural Networks*, vol. 4, no. 2, pp. 251-257.

Hou, J., Dai, A. and Niebner, M., 2019. 3D-SIS: 3D semantic instance segmentation of RGB-D scans, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4421-4430.

Huang, Q., Wang, W. and Neumann, U., 2018. Recurrent slice networks for 3D segmentation of point clouds, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2626-2635.

Janai, J., Guney, F., Behl, A. and Geiger, A., 2017. Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art, *arXiv: 1704.05519*.

Kanezaki, A., Matsushita, Y. and Nishida, Y., 2018. RotationNet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5010-5019.

Kingma, D. P. and Ba, J., 2014. Adam: A method for stochastic optimization, *arXiv: 1412.6980v9*.

Kipf, T. N. and Welling, M., 2016. Semi-supervised classification with graph convolutional networks, *arXiv:1609.02907*.

Klokov, R. and Lempitsky, V., 2017. Escape from cells: Deep Kd-Networks for the recognition of 3D point cloud models, *Proceedings of International Conference on Computer Vision*, pp. 863-872.

Krahenbuhl, P. and Koltun, V., 2011. Efficient inference in fully connected CRFs with gaussian edge potentials, *Proceedings of Neural Information Processing Systems*, pp. 109-117.

Ku, J., Mozifian, M., Lee, J., Harakeh, A. and Waslander, S. L., 2018. Joint 3D proposal generation and object detection from view aggregation, *Proceedings of International Conference on Intelligent Robots and Systems*, pp. 1-8.

Kumar, B., Pandey, G., Lohani, B. and Misra, S. C., 2019. A multi-faceted CNN architecture for automatic classification of mobile LiDAR data and an algorithm to reproduce point cloud samples for enhanced training, *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 147, pp. 80-89.

Kundu, A., Li, Y. and Rehg, J. M., 2018. 3D-RCNN: Instance-level 3D object reconstruction via render-and-compare, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3559-3568.

Lahoud, J. and Ghanem, B., 2017. 2D-driven 3D object detection in RGB-D images, *Proceedings of International Conference on Computer Vision*, pp. 4632-4640.

Lan, S., Yu, R., Yu, G. and Davis, L. S., 2019. Modeling local geometric structure of 3D point clouds using geo-CNN, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 998-1008.

Landrieu, L. and Simonovsky, M., 2018. Large-scale point cloud semantic segmentation with superpoint graphs, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4558-4567.

Le, T. and Duan, Y., 2018. PointGrid: A deep network for 3D shape understanding, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9204-9214.

Lecun, Y., Bengio, Y. and Hinton, G. E., 2015. Deep learning, *Nature*, vol. 521, no. 7553, pp. 436-444.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J. Z., Langer, D., Pink, O. and Pratt, V., 2011. Towards fully autonomous driving: Systems and algorithms, *Proceedings of the IEEE Intelligent Vehicles Symposium*, pp. 163-168.

Li, B., 2017. 3D fully convolutional network for vehicle detection in point cloud, *Proceedings of the International Conference on Intelligent Robots and Systems*, pp. 1513-1518.

Li, B., Zhang, T. and Xia, T., 2016. Vehicle detection from 3D Lidar using fully convolutional network, *arXiv: 1608.07916*.

Li, J., Bi, Y. and Lee, G. H., 2019. Discrete rotation equivariance for point cloud recognition, *arXiv: 1904.00319*.

Li, L., Huang, W.-L., Liu, Y., Zheng, N.-N. and Wang, F.-Y., 2016. Intelligence testing for autonomous vehicles: A new approach, *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 2, pp. 158-166.

Li, M., Hu, Y., Zhao, N. and Guo, L., 2019. LPCCNet: A lightweight network for point cloud classification, *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 6, pp. 962-966.

Li, P., Chen, X. and Shen, S., 2019. Stereo R-CNN based 3D object detection for autonomous driving, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7644-7652.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X. and Chen, B., 2018. PointCNN: Convolution on χ-transformed points, *arXiv: 1801.07791*.

Li, Y., Ma, L., Tan, W., Sun, C., Cao, D. and Li, J., 2020. GRNet: Geometric relation network for 3D object detection from point clouds, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 165, pp. 43-53.

Li, Y., Ma, L., Zhong, Z., Cao, D. and Li, J., 2020. TGNet: Geometric graph CNN on 3D point cloud segmentation, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 5, pp. 3588-3600.

Liang, M., Yang, B., Chen, Y., Hu, R. and Urtasun, R., 2019. Multi-task multi-sensor fusion for 3D object detection, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7345-7353.

Liang, M., Yang, B., Wang, S. and Urtasun, R., 2018. Deep continuous fusion for multi-sensor 3D object detection, *Proceedings of the European Conference on Computer Vision*, pp. 663-678.

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X. and Pietikäinen, M., 2020. Deep learning for generic object detection: A survey, *International Journal of Computer Vision*, vol. 128, no. 2, pp. 261-318.

Liu, Y., Fan, B., Xiang, S. and Pan, C., 2019. Relation-shape convolutional neural network for point cloud analysis, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8895-8904.

Liu, Y., Wang, C., Song, Z. and Wang, M., 2018. Efficient global point cloud registration by matching rotation invariant features through translation search, *Proceedings of the European Conference on Computer Vision*, pp. 460-474.

Long, J., Shelhamer, E. and Darrell, T., 2015. Fully convolutional networks for semantic segmentation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431-3440.

Luo, Z., Li, J., Xiao, Z., Mou, Z. G., Cai, X. and Wang, C., 2019. Learning high-level features by fusing multi-view representation of MLS point clouds for 3D object recognition in road environments, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 150, pp. 44-58.

Ma, L., Li, Y., Li, J., Wang, C., Wang, R. and Chapman, M. A., 2018. Mobile laser scanned point-clouds for road object detection and extraction: A review, *Remote Sensing*, vol. 10, no. 10, pp. 1531.

Masci, J., Boscaini, D., Bronstein, M. M. and Vandergheynst, P., 2015. Geodesic convolutional neural networks on riemannian manifolds, *Proceedings of the International Conference on Computer Vision*, pp. 832-840.

Maturana, D. and Scherer, S., 2015. VoxNet: A 3D convolutional neural network for real-time object recognition, *Proceedings of IEEE International Workshop on Intelligent Robots and Systems*, pp. 922-928.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S. and Geiger, A., 2019. Occupancy networks: Learning 3D reconstruction in function space, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4460-4470.

Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J. and Bronstein, M. M., 2017. Geometric deep Learning on graphs and manifolds using mixture model CNNs, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5425-5434.

Mousavian, A., Anguelov, D., Flynn, J. and Kosecka, J., 2017. 3D bounding box estimation using deep learning and geometry, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5632-5640.

Nguyen, A. and Le, B., 2013. 3D point cloud segmentation: A survey, *Proceedings of the IEEE Conference on Robotics Automation and Mechatronics*, pp. 225-230.

Pham, Q., Nguyen, T., Hua, B., Roig, G. and Yeung, S., 2019. JSIS3D: Joint semantic-instance segmentation of 3D point clouds with multi-task pointwise networks and multi-value conditional random fields, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8827-8836.

Qi, C. R., Litany, O., He, K. and Guibas, L. J., 2019. Deep Hough voting for 3D object detection in point clouds, *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9277-9286.

Qi, C. R., Liu, W., Wu, C., Su, H. and Guibas, L. J., 2018. Frustum pointnets for 3D object detection from RGB-D data, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 918-927.

Qi, C. R., Su, H., Mo, K. and Guibas, L. J., 2017. Pointnet: Deep learning on point sets for 3D classification and segmentation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652-660.

Qi, C. R., Su, H., Niebner, M., Dai, A., Yan, M. and Guibas, L. J., 2016. Volumetric and multi-view CNNs for object classification on 3D data, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5648-5656.

Qi, C. R., Yi, L., Su, H. and Guibas, L. J., 2017. PointNet++: Deep hierarchical feature learning on point sets in a metric space, *Proceedings of the Conference on Advances in Neural Information Processing Systems*, pp. 5099-5108.

Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster R-CNN: Towards real-time object detection with region proposal networks, *Proceedings of the Conference on Advances in Neural Information Processing Systems*, vol. 2015, pp. 91-99.

Ren, Z. and Sudderth, E. B., 2016. Three-dimensional object detection and layout prediction using clouds of oriented gradients, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1525-1533.

Ren, Z. and Sudderth, E. B., 2018. 3D object detection with latent support surfaces, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 937-946.

Rethage, D., Wald, J., Sturm, J., Navab, N. and Tombari, F., 2018. Fully-convolutional point networks for large-scale point clouds, *Proceedings of the European Conference on Computer Vision*, pp. 596-611.

Riegler, G., Ulusoy, A. O. and Geiger, A., 2017. OctNet: Learning deep 3D representations at high resolutions, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6620-6629.

Roynard, X., Deschaud, J. E. and Goulette, F., 2018. Classification of point cloud scenes with multiscale voxel deep network, *arXiv: 1804.03583*.

Sauder, J. and Sievers, B., 2019. Context prediction for unsupervised deep learning on point clouds, *arXiv: 1901.08396v1*.

Shen, Y., Feng, C., Yang, Y. and Tian, D., 2018. Mining point cloud local structures by kernel correlation and graph pooling, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4548-4557.

Shi, S., Wang, X. and Li, H., 2019. PointRCNN: 3D object proposal generation and detection from point cloud, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-779.

Shoef, M., Fogel, S. and Cohenor, D., 2019. PointWise: An unsupervised point-wise feature learning network, *arXiv: 1901.04544*.

Simonovsky, M. and Komodakis, N., 2017. Dynamic edge-conditioned filters in convolutional neural networks on graphs, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 29-38.

Song, S., Lichtenberg, S. P. and Xiao, J., 2015. SUN RGB-D: A RGB-D scene understanding benchmark suite, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 567-576.

Song, S. and Xiao, J., 2014. Sliding shapes for 3D object detection in depth images, *Proceedings of the European Conference on Computer Vision*, pp. 634-651.

Song, S. and Xiao, J., 2016. Deep sliding shapes for amodal 3D object detection in RGB-D images, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 808-816.

Story, M. and Congalton, R. G., 1986. Accuracy assessment: A user's perspective, *Photogrammetric Engineering and Remote Sensing,* vol. 52, no. 3, pp. 397-399.

Su, H., Maji, S., Kalogerakis, E. and Learnedmiller, E., 2015. Multi-view convolutional neural networks for 3D shape recognition, *Proceedings of the International Conference on Computer Vision*, pp. 945-953.

Tagliasacchi, A., Zhang, H. and Cohenor, D., 2009. Curve skeleton extraction from incomplete point cloud, *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, vol. 28, pp. 71.

Tatarchenko, M., Dosovitskiy, A. and Brox, T., 2017. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs, *Proceedings of the International Conference on Computer Vision*, pp. 2107-2115.

Tchapmi, L. P., Choy, C. B., Armeni, I., Gwak, J. and Savarese, S., 2017. SEGCloud: Semantic segmentation of 3D point clouds, *Proceedings of the International Conference on 3D Vision*, pp. 537-547.

Thomas, H., Goulette, F., Deschaud, J. and Marcotegui, B., 2018. Semantic classification of 3D point clouds with multiscale spherical neighborhoods, *Proceedings of the International Conference on 3D Vision*, pp. 390-398.

Wang, C., Samari, B. and Siddiqi, K., 2018. Local spectral graph convolution for point set feature learning, *Proceedings of the European Conference on Computer Vision*, pp. 56-71.

Wang, D. Z. and Posner, I., 2015. Voting for voting in online point cloud object detection, *Proceedings of the Conference on Robotics*, vol. 1, pp. 10-15607.

Wang, L., Huang, Y., Hou, Y., Zhang, S. and Shan, J., 2019. Graph attention convolution for point cloud semantic segmentation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10296-10305.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M. and Solomon, J., 2019. Dynamic graph CNN for learning on point clouds, *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 146.

Wang, Z. and Jia, K., 2019. Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection, *arXiv: 1903.01864*.

Wang, Z., Zhang, L., Fang, T., Mathiopoulos, P. T., Tong, X., Qu, H., Xiao, Z., Li, F. and Chen, D., 2015. A multiscale and hierarchical feature extraction method for terrestrial laser scanning point cloud classification, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 5, pp. 2409-2425.

Wen, C., Sun, X., Li, J., Wang, C., Guo, Y. and Habib, A., 2019. A deep learning framework for road marking extraction, classification and completion from mobile laser scanning point clouds, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 147, pp. 178-192.

Worrall, D. E. and Brostow, G. J., 2018. CubeNet: Equivariance to 3D rotation and translation, *Proceedings of the European Conference on Computer Vision*, pp. 585-602.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X. and Xiao, J., 2015. 3D ShapeNets: A deep representation for volumetric shapes, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1912-1920.

Xiang, Y., Choi, W., Lin, Y. and Savarese, S., 2015. Data-driven 3D voxel patterns for object category recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1903-1911.

Xie, S., Liu, S., Chen, Z. and Tu, Z., 2018. Attentional ShapeContextNet for point cloud recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4606-4615.

Xu, D., Anguelov, D. and Jain, A., 2018. PointFusion: Deep sensor fusion for 3D bounding box estimation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 244-253.

Xu, Y., Fan, T., Xu, M., Zeng, L. and Qiao, Y., 2018. SpiderCNN: Deep learning on point sets with parameterized convolutional filters, *Proceedings of the European Conference on Computer Vision*, pp. 87-102.

Yang, B., Luo, W. and Urtasun, R., 2018. PIXOR: Real-time 3D object detection from point clouds, *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7652-7660.

Yang, B., Wang, J., Clark, R., Hu, Q., Wang, S., Markham, A. and Trigoni, N., 2019. Learning object bounding boxes for 3D instance segmentation on point clouds, *Proceedings of the 33rd Conference Neural Information Processing Systems*, pp. 6740-6749.

Yang, Z., Sun, Y., Liu, S., Shen, X. and Jia, J., 2018. IPOD: Intensive point-based object detector for point cloud, *arXiv: 1812.05276*.

Yang, Z., Sun, Y., Liu, S., Shen, X. and Jia, J., 2019. STD: Sparse-to-dense 3D object detector for point cloud, *arXiv: 1907.10471*.

Ye, X., Li, J., Huang, H., Du, L. and Zhang, X., 2018. 3D recurrent neural networks with context fusion for point cloud semantic segmentation, *Proceedings of the European Conference on Computer Vision*, pp. 415-430.

Yew, Z. J. and Lee, G. H., 2018. 3DFeat-Net: Weakly supervised local 3D features for point cloud registration, *Proceedings of the European Conference on Computer Vision*, pp. 630-646.

Yi, L., Su, H., Guo, X. and Guibas, L. J., 2017. SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6584-6592.

Yi, L., Zhao, W., Wang, H., Sung, M. and Guibas, L. J., 2019. GSPN: Generative shape proposal network for 3D instance segmentation in point cloud, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3947-3956.

You, H., Feng, Y., Ji, R. and Gao, Y., 2018. PVNet: A joint convolutional network of point cloud and multi-view for 3D shape recognition, *Proceedings of the 26th ACM International Conference on Multimedia*, pp. 1310-1318.

Zhang, L. and Zhang, L., 2018. Deep learning-based classification and reconstruction of residential scenes from large-scale point clouds, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 4, pp. 1887-1897.

Zhao, H., Jiang, L., Fu, C. and Jia, J., 2019. PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5565-5573.

Zheng, S., Jayasumana, S., Romeraparedes, B., Vineet, V., Su, Z., Du, D., Huang, C. and Torr, P. H. S., 2015. Conditional random fields as recurrent neural networks, *Proceedings of the International Conference on Computer Vision*, pp. 1529-1537.

Zhong, Y., Han, X. and Zhang, L., 2018. Multi-class geospatial object detection based on a position-sensitive balancing framework for high spatial resolution remote sensing imagery, *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 138, pp. 281-294.

Zhong, Z., 2019. Spectral-spatial neural networks and probabilistic graph models for hyperspectral image classification, *http://hdl.handle.net/10012/14893*.

Zhou, Y. and Tuzel, O., 2018. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection, *Proceedings of the International Conference on Computer Vision*, pp. 4490-4499.

Appendix A. List of Publications during Ph.D. Study

Refereed Journal Papers

- **Li, Y**., Ma, L.**,** Zhong, Z., Liu, F., Cao, D. and Li, J. 2020. Deep Learning for LiDAR point clouds in autonomous driving: A review. IEEE Transactions on Neural Networks and Learning Systems, doi:10.1109/TNNLS.2020.3015992. (IF=8.793)
- **Li, Y**., Ma, L.**,** Tan, W., Sun, C., Cao, D., Li, J. 2020. GRNet: Geometric relation network for 3D object detection from point clouds, ISPRS Journal of Photogrammetry and Remote Sensing, vol. 165, pp. 43-53 (IF=7.319)
- **Li, Y**., Ma, L.**,** Zhong, Z., Cao, D. and Li, J. 2020. TGNet: Geometric graph CNN on 3D point cloud segmentation. IEEE Transactions on Geoscience and Remote Sensing, vol. 58, no. 5, pp. 3588-3600 (IF=5.855)
- Ma, L., **Li, Y.**, Li, J., Wang, C., Wang, R. and Chapman, M.A. Mobile laser scanned point-clouds for road object detection and extraction: A review. Remote Sensing, vol. 10, no. 10, pp.1531, 2018. (equally contributed with the first author) (IF=4.509)
- Ma, L., **Li, Y**., Li, J., Yu, Y., Marcato, J., Goncalves, W., Chapman, M., 2020. Capsule-based networks for road marking extraction and classification from mobile LiDAR point clouds, IEEE Transactions on Intelligent Transportation Systems, doi:10.1109/TITS.2020.2990120 (IF=6.319)
- Sun, C., Vianney, J. M., **Li, Y**., Chen, L., Li, L., Wang, F., Cao, D., 2020. Proximity based automatic data annotation for autonomous driving. IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 2, pp. 395-404. (IF=5.129)
- Ma, L., **Li, Y**., Li, J., Tan, W., Yu, Y., Chapman, M., 2020. Multi-scale point-wise convolutional neural networks for 3D object segmentation from LiDAR point clouds in large-scale urban environments, IEEE Transactions on Intelligent Transportation Systems, DOI: 10.1109/TITS.2019.2961060. (IF=6.319)
- Ma, L., **Li, Y.**, Li, J., Zhong, Z. and Chapman, M.A. Generation of horizontally curved driving lines in HD maps using mobile laser scanning point clouds. IEEE Journal of

Selected Topics in Applied Earth Observations and Remote Sensing, vol. 12, no. 5, pp.1572-1586, 2019. (IF=3.827)

Refereed Conference Papers

- **Li, Y.**, Ma, L., Huang, Y., Li, J. 2018. Segment-based traffic sign detection from mobile laser scanning data. IEEE International Geoscience and Remote Sensing Symposium, pp. 4607-4610.
- Tan, W., Qin, N., Ma, L., **Li, Y.**, Du, J., Cai, G., Li, J., 2020. Toronto-3D: A large-scale mobile LiDAR dataset for semantic segmentation of urban roadways, IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 202-211.
- Ma, L., Chen, Z., **Li, Y**., Zhang, D., Li, J., Chapman, M., 2019. Multispectral airborne laser scanning point-clouds for land cover classification using convolutional neural networks, ISPRS Geospatial Week 2019, ISPRS Archive, XLII-2/W13, 79-86.