TGNet: Geometric Graph CNN on 3-D Point Cloud Segmentation

Ying Li, Lingfei Ma, *Student Member, IEEE*, Zilong Zhong, *Student Member, IEEE*, Dongpu Cao, *Member, IEEE*, and Jonathan Li, *Senior Member, IEEE*

***Abstract*— Recent geometric deep learning works define convolution operations in local regions and have enjoyed remarkable success on non-Euclidean data, including graph and point clouds. However, the high-level geometric correlations between the input and its neighboring coordinates or features are not fully exploited, resulting in suboptimal segmentation performance. In this article, we propose a novel graph convolution architecture, which we term as Taylor Gaussian mixture model (GMM) network (TGNet), to efficiently learn expressive and compositional local geometric features from point clouds. The TGNet is composed of basic geometric units, TGConv, that conduct local convolution on irregular point sets and are parametrized by a family of filters. Specifically, these filters are defined as the products of the local point features and the neighboring geometric features extracted from local coordinates. These geometric features are expressed by Gaussian weighted Taylor kernels. Then, a parametric pooling layer aggregates TGConv features to generate new feature vectors for each point. TGNet employs TGConv on multiscale neighborhoods to extract coarse-to-fine semantic deep features while improving its scale invariance. Additionally, a conditional random field (CRF) is adopted within the output layer to further improve the segmentation results. Using three point cloud data sets, qualitative and quantitative experimental results demonstrate that the proposed method achieves 62.2% average accuracy on ScanNet, 57.8% and 68.17% mean intersection over union (mIoU) on Stanford Large-Scale 3D Indoor Spaces (S3DIS) and Paris-Lille-3D data sets, respectively.**

***Index Terms*— Deep learning, LiDAR point clouds, semantic segmentation.**

I. Introduction

3-D point clouds acquired by LiDAR sensors have been widely applied in the field of remote sensing, computer vision, and autonomous driving specific for multiple tasks, e.g., 3-D objects classification [1], [2], detection [3], [4], and semantic segmentation [5], [6]. As one of these critical tasks, semantic segmentation is urgently desired for a comprehensive scene understanding. Similar to per-pixel image labeling, 3-D semantic segmentation seeks to attribute a semantic classification label to each 3-D-point. Given the features are hierarchically learned in an end-to-end trainable framework [7], deep convolutional neural network (CNN) models has achieved remarkable success in 2-D semantic segmentation tasks. However, compared with 2-D regular imagery data, 3-D point clouds are uneven, unstructured, noisy, and irregular data, which cannot exploit the classical 2-D 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 [8], 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 [9]. 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 [10]. 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 [8] and 2) the spectral filtering is calculated based on the whole input data, which requires high computation capability.

Thus, Masci *et al.* [11] proposed the first spatial-based CNN on non-Euclidean data, applying filters to local neighbors represented in geodesic polar coordinates. Qi *et al.* [3] constructed spatial-based CNNs by defining convolution kernels in local neighbors with respect to local Euclidean positional relationships between points. Monti *et al.* [12] 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 [8]. 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.

1) The high-level geometric correlations between the input and its neighboring coordinates or features are not

Manuscript received September 29, 2019; revised November 1, 2019; accepted December 3, 2019. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN-50503-10284 and the National Natural Science Foundation of China under Grant 41871380. *(Corresponding author: Jonathan Li.)*

Y. Li and L. Ma are with the Department of Geography and Environmental Management, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: y2424li@uwaterloo.ca; l53ma@uwaterloo.ca).

Z. Zhong is with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 1010, China (e-mail: z26zhong@uwaterloo.ca).

D. Cao is with the Waterloo Cognitive Autonomous Driving Lab, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: dongpu.cao@uwaterloo.ca).

J. Li is with the Department of Geography and Environmental Management, University of Waterloo, Waterloo, ON N2L 3G1, Canada, and also with the Department of System Design Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: junli@uwaterloo.ca).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2019.2958517

fully exploited in defining convolution kernels. These correlations can enhance the kernel's shape description capability.

2) The traditional aggregation functions, e.g., max or mean, discard or neglect the structural connection among local neighbors because different neighbors 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 neighbors extracted from point clouds for semantic segmentation. These filters are defined as products of local neighbor 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 neighbors, 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 neighbor 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 neighborhoods 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. Compared with traditional approaches, the proposed algorithm possesses three advantages which are as follows.

1) We propose novel convolutional filters to capture local correlations described by neighborhood features and local geometric feature. Such a geometric feature is mapped from local coordinates using a family of Gaussian weighted Taylor functions and can enhance the filter's shape description capability.
2) We extract point features in a hierarchically multiscale way, which can ensure the information from the coarse-to-fine scale can be combined together to increase the segmentation performance.
3) We construct an end-to-end trainable network while adding a CRF layer after the output layer. Qualitative and quantitative experimental results on three public benchmarks demonstrate the effectiveness of the proposed method.

II. Related Works

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 [13]. 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.

A. 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 [14]–[17] are the most representative frameworks in existing 3-D Euclidean-structured deep models applied on large-scale point clouds.

The inputs of these methods are 3-D volumetric grids voxelized from the raw point clouds. In early voxel-based networks, convolution is operated in regular and uniform voxel grids [15]. 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, 3-D ShapeNets [15] inputted volumetric grids with size 30*30*30 into CNN architecture, the geometric 3-D shape was represented by binary variables with a probabilistic distribution of a 3-D voxel grid. Instead of limiting the size of the input volume, Kd-networks [16] adaptively divided the input data into hierarchical grids, which further reduce the computation cost and memory. OctNet [17] hierarchically splitted the 3-D 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 3-D shapes and surfaces, are not exploited.

B. 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 3-D tasks, such as segmentation [1], [3] and classification [16], [18].

1) Point Cloud-Based Models: Volumetric input of 3-D point clouds is still computational and complex, a more simpler network PointNet was proposed by Qi *et al.* [1] which takes point cloud directly as input. Symmetry function was used to handle unordered 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 been achieved in 3-D 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.* [3] to compensate the local feature extraction problem. This network was applied in raw input point clouds with various resolutions and assembled local features using a hierarchical architecture. PointCNN [19] 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 neighbors, which limit their performance in semantic segmentation.

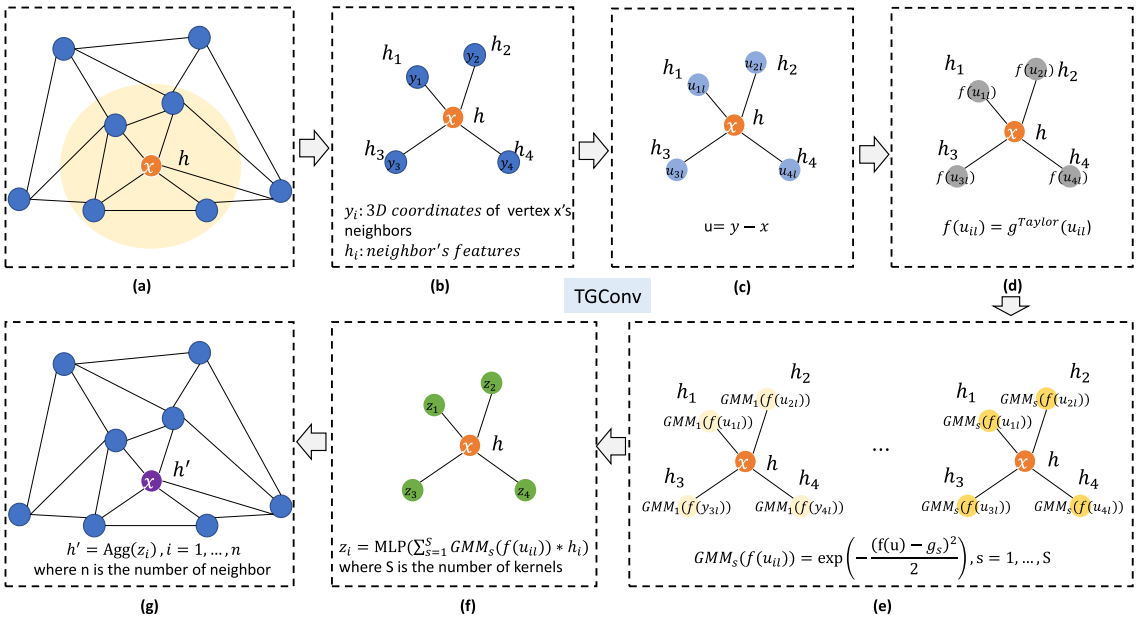


Fig. 1. TGConv on graph representation of point clouds: for each vertex x, given its KNNs coordinates y_i and features h_i, we conduct local operation in such non-Euclidean domain, by parametrizing a family of convolutional filters. These filters are defined as products of local neighbor point features with geometric features extracted from pseudocoordinates expressed by a family of Gaussian weighted Taylor kernel functions. Then they are aggregated via a parametrized pooling operation to new feature h' of vertex x. (a) Input. (b) Vertex x and its local graph. (c) Pseudocoordinate. (d) Geometric coordinate mapping. (e) Weight function. (f) Feature mapping. (g) Output-Aggregation.

2) 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 [20]. Yi *et al.* [9] 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 result in high computation cost. Thus, Wang *et al.* [21] carried out the graph convolution on the local point set and applied a recursive clustering and pooling operation to aggregate information from spectral-close nodes.

Spatial-based graph CNNs are commonly operated on groups of spatially close neighbors. In [22], features from local neighborhoods 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.* [18] also constructed a local neighborhood graph to learn the local geometric features. The EdgeConv was applied on the edges connecting neighboring pairs of each point. Besides, the given fixed graph was dynamically updated to extract high level local spatial information. However, not all neighbors contribute equally. Wang *et al.* [23] introduced an attention scheme in graph-based point cloud segmentation by assigning specific attentional weights to different neighboring points. This operation can dynamically adapt the kernel to different objects with various structures.

In order to improve the expressiveness of convolutional kernels in a spatial domain, Monti *et al.* [12] applied a parametric construction of the convolution on local graph patches extracted from spatial-domain neighborhoods. They treated the patch operators as a function of a local graph and learned a set of functions based on a mixture of Gaussian kernels. In [24], a family of convolutional filters was parametrized by a Taylor polynomial to improve their geometric expressions. Their filter was defined as a product of a step function which extracts local geodesic information and a Taylor polynomial which increases the filter's expressiveness. Our filter is constructed based on the above two works, but we combine the Taylor functions and GMM weight functions to map the low-level coordinate information into a high-level geometric feature. Such an operation improves the performance of point cloud semantic segmentation.

C. CRF in Deep Learning

In order to refine the coarse CNN segmentation results, the CRF is commonly applied in many works [26]–[28]. The CRFs have the advantage of combining low-level information such as the interactions between points to output multiclass inference for multiclass per-point labeling tasks, which compensate the fine local details that CNNs fail to capture. For 3-D point cloud, following a CRF-recurrent neural network (RNN) [29], SegCloud [27] extends the implementation of CRF into 3-D point clouds after a fully CNN. However, as CRF is applied as an individual part following the CNNs, it is difficult to explore the power of the combination of CNNs and CRF.

III. Methodology

The goal of our network is to make a dense predicted semantic label for each input point, such as ground, tree, and building. This task can be concluded as: given a set

TABLE I
CHOICE OF PSEUDOCOORDINATES AND WEIGHT FUNCTIONS OF SEVERAL GEOMETRIC CNN MODELS

Method	Aggregation	Pseudo-coordinates	Pseudo-coordinates u(x,y)	Weight Function $w_j(\mathbf{u}), j = 1, \ldots, J$
PointNet++ [3]	max	Local Euclidean	$u(x,y) = u(y) - u(x)$	-
MoNet [12]	$\sum$	Vertex degree	$u(x,y) = \left(\frac{1}{\sqrt{\deg(x)}}, \frac{1}{\sqrt{\deg(y)}}\right)^\top$	$w_j(u) = \exp\left(-\frac{1}{2}\left(u - \boldsymbol{\mu}_j\right)^\top \boldsymbol{\Sigma}_j^{-1}\left(u - \boldsymbol{\mu}_j\right)\right)$
GCN [25]	$\sum$	Vertex degree	deg(x), deg(y)	$\left(1 - \left\|1 - \frac{1}{\sqrt{u_1}}\right\|\right)\left(1 - \left\|1 - \frac{1}{\sqrt{u_2}}\right\|\right)$
DGCNN [18]	max	Local Euclidean	$u(x,y) = u(y) - u(x)$	-

of 3-D points $P = \{p_1, p_2, p_i, \ldots, p_n\} \in R^3$ and a candidate label set $L = \{l_1, l_2, \ldots, l_k\}$, assign each input point p_i with one of the k semantic labels. Thus, in this section, we first give the preliminary knowledge of geometric convolution, which constitutes the important basis of our method. Then, we introduce our TGConv in detail. Finally, an end-to-end point cloud segmentation framework TGNet with our proposed TGConv is presented.

A. 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 2-D imagery convolution in regular Euclidean domain, per-pixel patch extraction at each position is always the same. However, due to the unstructured and unordered 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.

1) The local patch extraction should be shift-invariant, however, it is actually position-dependent.
2) 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.* [12] and Kipf and Welling [25] constructed patch operators $D(\cdot)$ by defining a family of learnable weight functions $w_1(u), \ldots, w_J(u)$ of local patch (e.g., local graph) represented by pseudocoordinates u. Given vertex x and its neighbor [denoted as $x' \in \mathcal{N}(x)$] features f, the patch operator can be formulated as the weighted summation of f

$$D(x)f = \sum_{x' \in \mathcal{N}(x)} f(x') w_j(u(x, x')), \quad j = 1, \ldots, J. \quad (1)$$

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

$$(f \star g)(x) = \sum_{j=1}^{J} g_\theta (D_j(x) f) \quad (2)$$

where $\star$ 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 GCN [25] and mixture model network (MoNet) [12] 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 neighbor 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.

B. TGConv

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{U})$ constructed from a given 3-D point cloud $P = \{p_1, \ldots, p_n\} \subseteq R^3$ according to their spatial neighbors, where $\mathcal{V} = \{1, 2, \ldots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represent the set of vertices and edges, respectively, and $\mathbf{U}$ contains 3-D pseudocoordinates $u(x, y) \subseteq R^3$ for each directed edge $(x, y) \in \mathcal{E}$. Denote each point $y \in \mathcal{N}(x)$ as the neighbor set of vertex x, $u(x, y)$ is a 3-D vector of pseudocoordinates for each y. Let $\boldsymbol{h} = \{h_1, h_2, \ldots, h_N\}$ be a set of input vertex features, each feature $h_i \in \mathbb{R}^F$ is associated with a corresponding graph vertex $i \subseteq V$, where F is the feature dimension of each vertex.

To leverage spatially local correlation, we mimic (1) and (2) to conduct local operations on the local graph, by parametrizing a family of convolutional filters. These filters are defined as products of local neighbor point features with geometric features extracted from local coordinates expressed by a family of Gaussian weighted Taylor kernel functions [see Fig. 1]. Then they are aggregated via a parametric pooling operation to new point set features $\boldsymbol{h}' = \{h'_1, h'_2, \ldots, h'_N\}$ with $h'_i \in \mathbb{R}^K$.

In (1), the patch operator is defined directly on the pseudocoordinates $u(x, y)$. Although geometric information can be extracted, however, high-level geometric spatial features are not exploited. Thus, we map the pseudocoordinates to a high-level geometric feature using a function $T(\mathbf{u})$:$R^3 \to R$, which 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

$$(f \star g)(x) = \mathbf{Agg}\left(g_\theta\left(\sum_{s=1}^{S} D_s(\mathbf{x}) h_y\right)\right), \quad y \in \mathcal{N}(x) \quad (3)$$

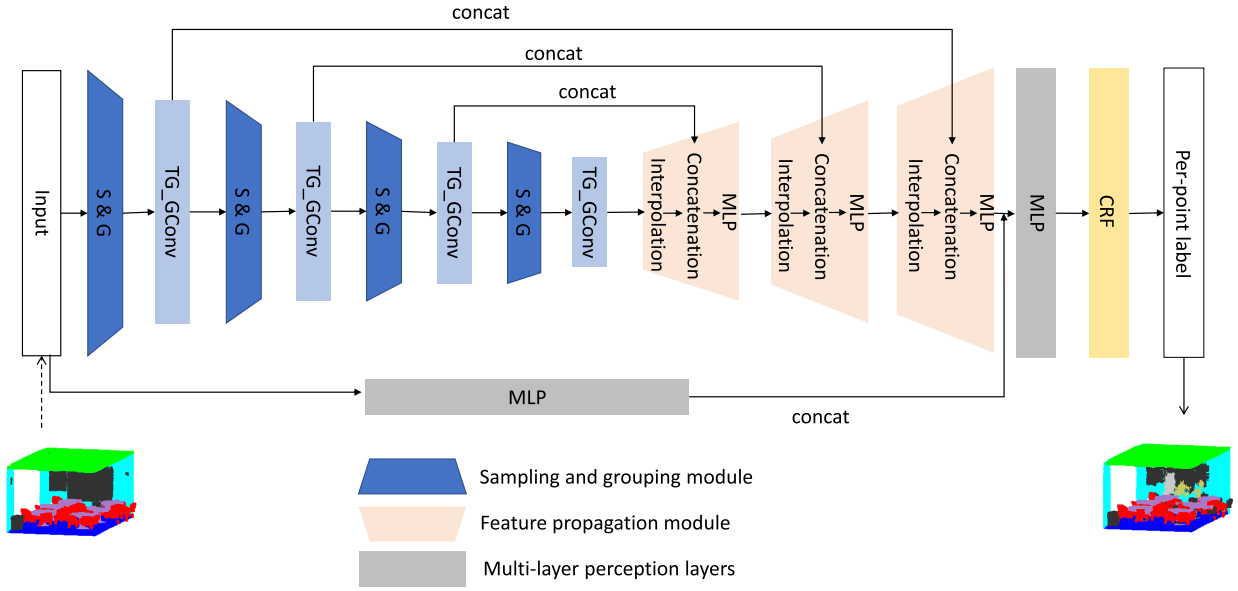


Fig. 2. Framework of our TGNet. It employs a multiscale hierarchical architecture by operating TGConv on neighbors at multiple scales, which allows it to extract coarse-to-fine semantic deep features. Besides, a shared MLP is applied to the raw input to extract per-point features. These learned features are combined with the interpolated features learned from the finest layer to predict per-point semantic label likelihood. Finally, a CRF layer is combined within the output layer to further improve the segmentation result.

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

$$D_s(\mathbf{x}) = w_s(T(\mathbf{u})), \quad s = 1, \ldots, \mathcal{S} \tag{4}$$

with $\mathbf{u}$ representing the pseudocoordinates, $\mathbf{w}(T) = (w_1(T), \ldots, w_S(T))$ 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 pseudocoordinates $\mathbf{u}$, geometric pseudocoordinates mapping function $T(\mathbf{u})$, weight functions $\mathbf{w}(T)$, feature mapping function $g_\theta(\cdot)$, and aggregation function $\mathbf{Agg}(\cdot)$.

1) Pseudocoordinates: Pseudocoordinates, such as polar, spherical, or Cartesian coordinates, encode local positional relationships between points [10] and can be used to describe local geometric features. Table I lists the selection of pseudocoordinates $\mathbf{u}$ and weight function $\mathbf{w}(\mathbf{u})$ of some geometric deep learning methods [3], [12], [18], [25]. For example, MoNet [12] and GCN [25] define their kernels on the pseudocoordinates based on the degree of graph vertices. PointNet++ [3] and DGCNN [18] select the local Euclidean coordinates as their pseudocoordinates. In order to reduce the computation cost and exploit the original geometric feature from 3-D coordinates, local Euclidean coordinates are selected as our pseudocoordinates. For each vertex x and vertex $y \in \mathcal{N}(x)$ in the neighborhood of x, we consider local pseudocoordinates $\mathbf{u}(x, y)$ as

$$\mathbf{u}(x, y) = y - x = (\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3})^T \tag{5}$$

where each vertex x is represented by 3-D Cartesian coordinates, and $(\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3})^T$ represent the corresponding pseudocoordinate along each axis of each neighborhood point y to point x.

2) Geometric Pseudocoordinates Mapping Function: The pseudocoordinates leverage only the low-level spatial information, and the high-level structural and geometric information among pseudocoordinates is not exploited. Based on that, we design our filters considering the high-level structural information of pseudocoordinates 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 $T(\mathbf{u})$ is used to leverage the intrinsic information among $(\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3})^T$ into high-level representation $g(\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3})$.

There are two important considerations for choosing such a mapping function $T(\mathbf{u})$: 1) the optimization is convenient to conduct and 2) it can interpolate arbitrary values in local graph. Inspired by SpiderCNN [24], the order-3 Taylor expansions of 3-D coordinates are used to map the local pseudocoordinates $\mathbf{u}$ into high-level geometric attribute

$$\begin{aligned} T(\mathbf{u}) = g_{\sigma^T}^{\text{Taylor}}(\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3}) = \sigma_0^T \\ + \sum_{i=1}^{3} \sigma_i^T \mathbf{u}_i \\ + \sum_{i=1}^{3} \sum_{j=1}^{3} \sum_{i \le j} \sigma_{ij}^T \mathbf{u}_i \mathbf{u}_j + \sum_{i=1}^{3} \sum_{i=1}^{3} \sum_{i \le j} \sum_{\substack{k=1 \\ \le k}}^{3} \sigma_{ijk}^T \mathbf{u}_i \mathbf{u}_j \mathbf{u}_k \end{aligned} \tag{6}$$

where σ_0^T, σ_i^T, σ_{ij}^T, and σ_{ijk}^T are all 1×1 learnable parameters. By varying these parameters, $g(\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3})$ can approximate arbitrary values.

3) Weight Function: In [12], a GMM is used as the weight functions $\mathbf{w}(\mathbf{u})$ with learnable parameters as

$$w_s(\mathbf{u}) = \exp\left(-\frac{1}{2}(\mathbf{u} - \boldsymbol{\mu}_s)^\top \boldsymbol{\Sigma}_s^{-1}(\mathbf{u} - \boldsymbol{\mu}_s)\right) \tag{7}$$

where $\mathbf{\Sigma}_s$ and μ_s are learnable $d \times d$ and $d \times 1$ covariance matrix and mean vector of a Gaussian kernel, respectively. Their experimental results have demonstrated that they can learn the geodesic features of non-Euclidean data with 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

$$w_s(\mathbf{u}) = \exp\left(-\frac{1}{2}(\mathbf{g} - \boldsymbol{g}_s)^2\right), \quad s = 1, \ldots, \mathcal{S} \tag{8}$$

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

$$h'_m = \sum_{s=1}^{S} \exp\left(-\frac{1}{2}\left(\mathbf{g}_{\sigma\mathbf{T}}^{\text{Taylor}}(\mathbf{u}) - \boldsymbol{g}_s\right)^2\right) h. \tag{9}$$

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

4) 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 [30]. Besides, MLP retains the crucial characteristic of standard convolution in grid domain: weight sharing. Thus, the input intermediate feature h'_m is mapped as

$$h'_\theta = MLP\left(h'_m\right). \tag{10}$$

5) 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 [3], 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:

$$h'_x = \max\left\{h'_{\theta y}\right\} + \frac{\sum_{j=1}^{k} \theta_j w_j h'_{\theta y}}{\sum_{j=1}^{k} w_j}, \quad y \in \mathcal{N}(x) \tag{11}$$

where

$$w_j = \frac{1}{d(p_y, p_x)^p} \tag{12}$$

and p_y, and p_x represent the coordinates of neighbor point y and the vertex x. k represents the number of neighbors. θ_j is a learnable 1×1 vector, which is used to learn most relevant neighbor features and reweight the nonrelevant neighbor 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 neighbors' features to influence prediction.

C. 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 Fig. 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++ [3], 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.

1) 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 [3] 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 $P_1, \ldots, P_L$, where L represents the number of scales. For each $P_l (l = 0, \ldots, L)$, a corresponding graph $\mathcal{G}_l = (\mathcal{V}_l, \mathcal{E}_l)$ can be constructed as described above.

Because our TGConv is operated in the local region for each vertex at multiple scales. Thus, how to find spatially important neighbors is of great significance. There are two ways to search the nearest neighbors: Spherical neighborhood [31] and K-nearest neighbor (KNN) [32]. The first one selects k neighbors randomly within radius γ, while KNN chooses the point with k smallest distance neighbors among all the input points. Thus, spherical neighborhood is adaptive to density variation. Because the point clouds are commonly distributed unevenly, the spherical neighborhood is selected to enhance the framework's invariance to density change.

We determine the radius of the local spherical neighbor experimentally. Our method processes both indoor and outdoor scenes. However, points in indoor scenes scanned by RGB-D camera have uniform point distribution, while points in outdoor scenes scanned by a mobile laser scanner (MLS) have irregular and sparse point density. Fixed radius is simple and

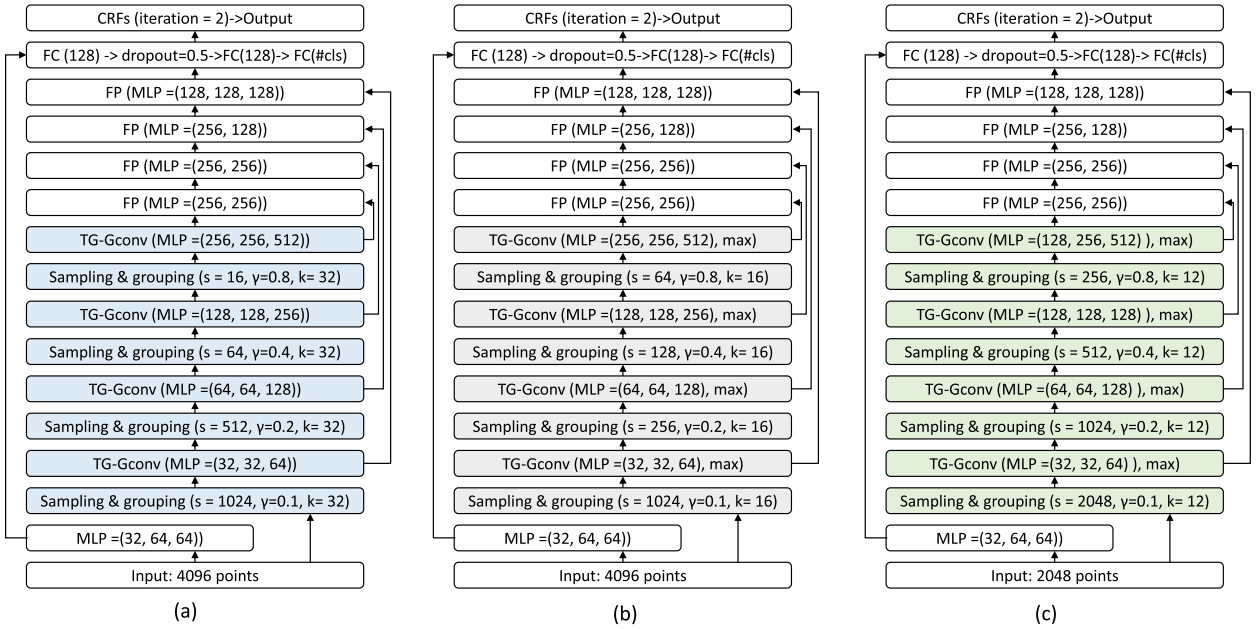


Fig. 3. TGNet model zoo for three different data sets. (a) ScanNet framework. (b) S3DIS framework. (c) Paris-Lille-3D framework. The input to the ScanNet and S3DIS data sets is 4096 points, while for the Paris-Lille-3D, the input point number is 2048 points. The main difference between these three frameworks is the sampling and grouping module with different subsampled ratio and nearest neighbors. Owing to the computation limitation, the last two frameworks use the max pooling aggregation method instead of our proposed max and parametric weighted average pooling operator.

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 our each sampling radius to a fixed value experimentally.

2) 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 $\boldsymbol{h}_l$ 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 $p_i, i = 1, \ldots, k$) 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

$$h_{l-1} = \frac{\sum_{i=1}^{k} w_i h_l}{\sum_{i=1}^{k} w_i}, \quad \text{where } w_i(x) = \frac{1}{d(p, p_i)^q}. \tag{13}$$

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.

3) CRF Layer: CRF [29] is commonly applied to post-process 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 [33], an approximate inference method is proposed. It assumes independence between semantic label distributions $Q(\mathbf{X}) = \prod_i Q_i(x_i)$, and derives the update equation

$$Q_i^+(x_i = l) = \frac{1}{Z_i} \exp\left\{ -\psi_u(x_i) - \sum_{l' \in \mathcal{L}} \mu(l, l') \sum_{m=1}^{K} w^{(m)} \times \sum_{j \neq i} k^{(m)}(f_i, f_j) Q_j(l') \right\}. \tag{14}$$

Based on that, Zheng *et al.* [29] 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.

IV. Results and Discussion

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 [34], Stanford Large-Scale 3D Indoor Spaces (S3DIS) data set [35] and Paris-Lille-3D data set [36]. Before we conduct experiments on the above three data sets, some ablation studies of TGNet is first analyzed to demonstrate the effectiveness of our method.

A. Data Sets

1) ScanNet [34]: The ScanNet data set contains 1513 scans by using RGB-D video streaming in indoor environments,

TABLE II
ABLATION STUDIES ON SCANNET TEST SET

Ablation studies	avg(%)
Aggregation methods	
Base-model [3]	57.82
TGNet (with max pooling)	61.42
TGNet (with max+ parametric weighted average pooling)	**62.20**
CRFasRNN	
TGNet (without CRF)	59.66
CRF-RNN (1 iteration)	61.42
CRF-RNN (2 iteration)	**62.20**
CRF-RNN (5 iteration)	61.04
Number of nearest neighbors	
k=8	56.76
k=16	59.98
k=24	60.45
k=32	**62.20**

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.

2) S3DIS Data Set [35]: 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.

3) Paris-Lille-3D [36]: 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 mobile LiDAR 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.

B. Evaluation Metrics

On ScanNet data set, we adopted accuracy and unweighted average accuracy [3] as our evaluation metric, which is different from the overall accuracy (OA) used in PointWeb [42]. OA means the proportion of correctly classified points among all the input points [43], 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) [18], mean IoU (mIoU) of each class [18], 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.

C. Ablation Studies and Analysis

In order to verify the effectiveness of our proposed aggregation method, CRFasRNN, and determine the number of nearest neighbors, we conduct several ablation studies on ScanNet test data set [34] and show their results in Table II.

1) Ablation Test of Aggregation Methods: Our base model is PointNet++ [3], which achieves 57.82% 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 0.78% 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 4.38% 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 [23], 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 [29] for the pairwise potentials of CRF. The testing results on the ScanNet test data set are also provided in Table II for comparing convenience. We can see that, within the integration of CRFasRNN layer, the average accuracy of semantic segmentation result is improved about 1.76% compared with TGNet without CRFasRNN layer. With two iterations, the CRF-RNN 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 Neighbors: We also study the number of nearest neighbors k chosen in TGNet,

TABLE III
SEMANTIC VOXEL LABEL PREDICTION ACCURACY (%) ON SCANNET TEST SCENES

Method	of the scenes	ScanNet [34]	ScanComplete [37]	3DMV [38]	Recurrent Slice Networks [39]	PointNet [1]	FCPN [40]	PointNet++ [3]	Matterport3D [41]	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	94.1	88.6	96.3	97.8	92.6	96.6
cab	2.4	49.8	44.5	54.4	31.3	5	52.1	39.8	91.1	46.6
bed	2	62.4	65.7	69.5	56	18	65.9	80.7	60.6	81.1
chair	3.8	69.3	75.1	79.5	65	35.9	81.6	86	20.7	82.2
sofa	2.5	75.7	72.1	70.6	55.4	32.8	76	68.3	28.4	85.3
table	3.3	68.4	63.8	71.3	51	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	3.2	81	84.3	1	83.5
pic	0.2	3.4	10.4	4.2	1	0	1.8	0.0	7.5	0.0
cntr	0.6	32.1	31.4	20	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	42.2
curt	0.7	7	49.8	15.2	6.8	0	6.1	48.7	85.4	69.3
fridg	0.3	66.4	38.7	59.9	37.9	0	54.7	54.7	6.8	60.6
show	0.04	46.8	46.8	57.3	29.9	0	48	85	20.2	84.9
toil	0.2	69.9	72.2	78.7	54.2	0	86.7	84.8	5.1	89.4
sink	0.2	39.4	47.4	48.8	34.8	0	53.5	62.8	27.5	70.6
bath	0.2	74.3	85.1	87	49.4	0.2	79.1	86.1	18.3	89.4
other	2.9	19.5	26.9	20.6	19	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**

where the results are provided in Table II. The number of nearest-neighbors 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 neighbors. The higher number of nearest neighbors is not tested due to the limitation of our computation capability.

D. Segmentation Results

1) 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 Fig. 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 neighbors 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 III gives 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 [1], [34], [37]–[41], our approach learns geometry features hierarchically. This is crucial for understanding scenes at different scales and labeling objects with different sizes. Although PointNet++ [3] learns hierarchical and geometry features at different scales, the geometric coordinates is 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++ [3], we have tested their published code in our own computer on this data set and achieved 57.8% unweighted accuracy, shown in Table II. This can further demonstrate the effectiveness of our method.

2) Semantic Segmentation on S3DIS: Although there are six labeled indoor areas in S3DIS data set, for a principled evaluation, we follow [1], [18], [23], [27], [44], [45] to choose Area 5 as our testing set and train our TGNet on the rest. 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 neighbors 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 IV. We can see that our TGNet achieves the best OA than other competitive methods [1], [18], [23], [27], [44], [45]. As the convolution weights of TGConv are assigned according to not only the spatial positions

TABLE IV
OA (%) AND MIOU (%) ON S3DIS DATA SET (TESTING ON AREA 5 AND TRAINING ON THE REST)

Method	OA	mIoU	ceiling	floor	wall	beam	column	window	door	chair	table	bookcase	sofa	board	clutter
PointNet [1]	-	41.1	88.8	97.3	69.8	0.1	3.9	46.3	10.8	52.6	58.9	40.3	5.9	26.4	33.4
SegCloud [27]	-	48.9	90.1	96.1	69.9	0.0	18.4	38.5	23.1	75.9	70.4	58.4	40.9	13.0	41.6
3P-RNN [44]	-	53.4	95.2	98.6	77.4	0.8	9.8	52.7	27.9	76.8	78.3	58.6	27.4	39.1	51.0
SPG [45]	86.4	58.0	89.4	96.9	78.1	0.0	42.8	48.9	61.6	84.7	75.4	69.8	52.6	2.1	52.2
DGCNN [18]	59.8	51.5	93.0	97.4	77.7	0.0	12.2	47.8	39.8	67.4	72.4	23.2	52.3	39.8	46.6
GACNet [23]	87.8	**62.9**	92.3	98.3	81.9	0.0	20.4	59.1	40.9	78.5	85.8	61.7	70.8	74.7	52.8
Ours	**88.5**	57.8	93.3	97.6	78.0	0.0	9.3	57.0	39.4	83.4	76.4	60.6	41.8	58.7	55.3

TABLE V
OA (%) AND MIOU (%) ON PARIS-LILLE-3D DATA SET

Method	OA	mIoU	Ground	Building	Pole	Bollard	Trash can	Barrier	Pedestrian	Car	Natural
PointNet [1]	93.88	40.19	97.5	91.32	26.47	6.26	8.99	8.53	4.36	74.26	44.03
PointNet++ [3]	88.67	36.07	91.98	79.38	27.89	27.57	0.35	5.43	1.82	67.95	22.30
DGCNN [18]	96.93	62.46	98.52	95.18	57.55	52.08	42.39	35.55	18.57	93.1	69.22
Ours	**96.97**	**68.17**	97.92	94.85	58.82	69.78	63.84	35.87	38.39	91.71	62.38

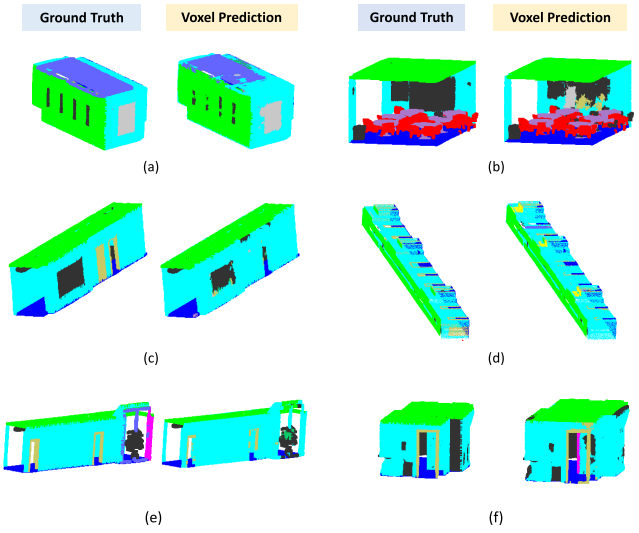


Fig. 4. Semantic segmentation results of S3DIS in six representative scenes with different room shapes and furniture. (a) Conference room. (b) Lobby. (c) Hallway 1. (d) Hallway 2. (e) Hallway 3. (f) WC.

but also the geometric attributes of the neighboring 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 GAT [23]. 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 Fig. 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.

3) 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 [1] and PointCNN [19], we first split the data set along the XOY plane and then sampled them into 5 m × 5 m blocks with a 0.1-m 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.

The sampling point in each layer is 2048, 1024, 512, 256, and the number of nearest neighbors 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 V. In general, our performance is on par with or better than other competitive algorithms for many classes. 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.

Fig. 5 shows comparison semantic segmentation results of DGCNN [18] and TGNet on Paris-Lille-3D data set. Compared with groundtruth, 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,

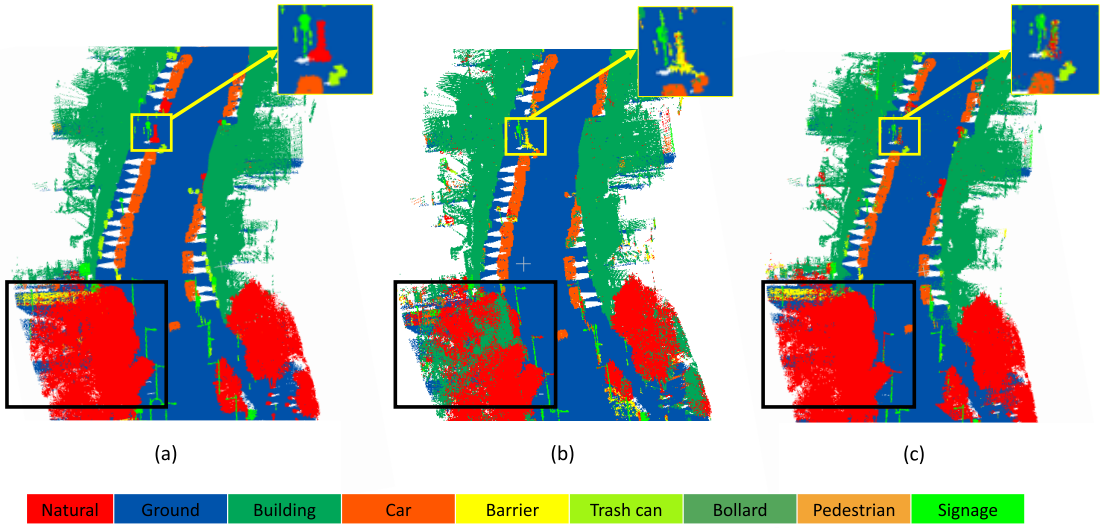


Fig. 5. Comparison semantic segmentation results of two algorithms on Paris-Lille-3D data set. (a) Ground-truth. (b) DGCNN [18]. (c) Ours.

TABLE VI
PARAMETER NUMBER AND RUNNING TIME COMPARISONS

Method	Parametres	Time	
		training	testing
TGNet (without CRF)	4.32M	0.058s	0.064s
TGNet	8.54M	0.064s	0.068s

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.

E. Optimizer, Model Size, Memory Usage, and Timing

The proposed method was implemented with Python 3.5 and TensorFlow 1.4 [46] on one GTX 1080ti GPU. We use ADAM optimizer [47] 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 VI. The model for ScanNet semantic voxel labeling with 4096 input points has 4.32M 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.

F. 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%) in Paris-Lille-3D MLS data set is much higher than that (88%) in S3DIS indoor data set. We conclude two main reasons for this difference: first, indoor scenes have strong occlusions and tight arrangements of common objects [48]; 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.

V. CONCLUSION

In this article, we address the 3-D point cloud segmentation problem in both indoor and outdoor environments. We have proposed a novel geometric graph convolution TGConv, which is defined as products of local neighbor point features with geometric features. 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 neighbors to improve TGConv performance in semantic segmentation. We also defined a parametrized pooling operation, composed of the max and a learnable distanced-based weight function for feature aggregation. 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 neighbors 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 have demonstrated that the proposed method achieves 62.2% average accuracy on ScanNet, 57.8% and 68.17% 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 3-D point clouds. However, our method still suffers one limitation in multiobject 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.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE CVPR*, 2017, pp. 652–660.

[2] Z. Wang *et al.*, "A multiscale and hierarchical feature extraction method for terrestrial laser scanning point cloud classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 5, pp. 2409–2425, May 2015.

[3] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. NIPS*, 2017, pp. 5099–5108.

[4] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-time 3D object detection from point clouds," in *Proc. IEEE CVPR*, Jun. 2018, pp. 7652–7660.

[5] A. Nina, P. Przemyslaw, H. Marco, K. Peter, and A. K. Skidmore, "Adaptive stopping criterion for top-down segmentation of als point clouds in temperate coniferous forests," *ISPRS J. Photogramm. Remote Sens.*, vol. 141, pp. 265–274, Jul. 2018.

[6] Z. Wang, L. Zhang, L. Zhang, R. Li, Y. Zheng, and Z. Zhu, "A deep neural network with spatial pooling (DNNSP) for 3D point cloud classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 8, pp. 4594–4604, Aug. 2018.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE CVPR*, Jun. 2015, pp. 3431–3440.

[8] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond Euclidean data," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 18–42, Jul. 2017.

[9] L. Yi, H. Su, X. Guo, and L. J. Guibas, "SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation," in *Proc. IEEE CVPR*, Jul. 2017, pp. 2282–2290.

[10] M. Fey, J. E. Lenssen, F. Weichert, and H. Müller, "SplineCNN: Fast geometric deep learning with continuous B-spline kernels," in *Proc. IEEE CVPR*, Jun. 2018, pp. 869–877.

[11] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst, "Geodesic convolutional neural networks on Riemannian manifolds," in *Proc. IEEE ICCV workshops*, Dec. 2015, pp. 37–45.

[12] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE CVPR*, Jul. 2017, pp. 5115–5124.

[13] E. Ahmed *et al.*, "Deep learning advances on different 3D data representations: A survey," 2018, *arXiv:1808.01462*. [Online]. Available: https://arxiv.org/abs/1808.01462

[14] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE CVPR*, Jun. 2018, pp. 4490–4499.

[15] Z. Wu *et al.*, "3D ShapeNets: A deep representation for, volumetric shapes," in *Proc. IEEE CVPR*, Jun. 2015, pp. 1912–1920.

[16] R. Klokov and V. Lempitsky, "Escape from cells: Deep Kd-networks for the recognition of 3D point cloud models," in *Proc. IEEE ICCV*, Oct. 2017, pp. 863–872.

[17] G. Riegler, A. O. Ulusoy, and A. Geiger, "OctNet: Learning deep 3D representations at high resolutions," in *Proc. IEEE CVPR*, Jul. 2017, pp. 3577–3586.

[18] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," 2018, *arXiv:1801.07829*. [Online]. Available: https://arxiv.org/abs/1801.07829

[19] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on X-transformed points," in *Proc. NIPS*, 2018, pp. 820–830.

[20] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*. [Online]. Available: https://arxiv.org/abs/1312.6203

[21] C. Wang, B. Samari, and K. Siddiqi, "Local spectral graph convolution for point set feature learning," in *Proc. ECCV*, 2018, pp. 52–66.

[22] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proc. IEEE CVPR*, Jul. 2017, pp. 3693–3702.

[23] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in *Proc. IEEE CVPR*, Jun. 2019, pp. 10296–10305.

[24] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," in *Proc. ECCV*, 2018, pp. 87–102.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: https://arxiv.org/abs/1609.02907

[26] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, 2001, pp. 282–289.

[27] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "segcloud: Semantic segmentation of 3D point clouds," in *Proc. DV*, 2017, pp. 537–547.

[28] L. Wang, Y. Huang, J. Shan, and L. He, "MSNet: Multi-scale convolutional network for point cloud classification," *Remote Sens.*, vol. 10, no. 4, p. 612, Apr. 2018.

[29] S. Zheng *et al.*, "Conditional random fields as recurrent neural networks," in *Proc. IEEE ICCV*, Dec. 2015, pp. 1529–1537.

[30] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Netw.*, vol. 4, no. 2, pp. 251–257, 1991.

[31] H. Thomas, F. Goulette, J.-E. Deschaud, and B. Marcotegui, "Semantic classification of 3D point clouds with multiscale spherical neighborhoods," in *Proc. DV*, 2018, pp. 390–398.

[32] F. Engelmann, T. Kontogianni, J. Schult, and B. Leibe, "Know what your neighbors do: 3D semantic segmentation of point clouds," in *Proc. ECCV*, 2018, pp. 395–409.

[33] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFS with Gaussian edge potentials," in *Proc. NIPS*, 2011, pp. 109–117.

[34] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE CVPR*, Jul. 2017, pp. 5828–5839.

[35] I. Armeni *et al.*, "3D semantic parsing of large-scale indoor spaces," in *Proc. IEEE CVPR*, Jun. 2016, pp. 1534–1543.

[36] X. Roynard, J.-E. Deschaud, and F. Goulette, "Classification of point cloud scenes with multiscale voxel deep network," 2018, *arXiv:1804.03583*. [Online]. Available: https://arxiv.org/abs/1804.03583

[37] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner, "ScanComplete: Large-scale scene completion and semantic segmentation for 3D scans," in *Proc. IEEE CVPR*, Jun. 2018, pp. 4578–4587.

[38] A. Dai and M. Nießner, "3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation," in *Proc. ECCV*, 2018, pp. 452–468.

[39] Q. Huang, W. Wang, and U. Neumann, "Recurrent slice networks for 3D segmentation of point clouds," in *Proc. IEEE CVPR*, Jun. 2018, pp. 2626–2635.

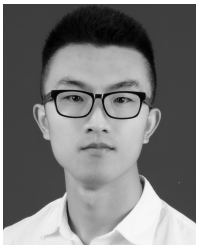
[40] D. Rethage, J. Wald, J. Sturm, N. Navab, and F. Tombari, "Fully-convolutional point networks for large-scale point clouds," in *Proc. ECCV*, 2018, pp. 596–611.

[41] A. Chang *et al.*, "Matterport3D: Learning from RGB-D data in indoor environments," 2017, *arXiv:1709.06158*. [Online]. Available: https://arxiv.org/abs/1709.06158

[42] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "PointWeb: Enhancing local neighborhood features for point cloud processing," in *Proc. IEEE CVPR*, Jun. 2019, pp. 5565–5573.
[43] M. Story and R. G. Congalton, "Accuracy assessment: A user's perspective," *Photogramm. Eng. Remote Sens.*, vol. 52, no. 3, pp. 397–399, 1986.
[44] X. Ye, J. Li, H. Huang, L. Du, and X. Zhang, "3D recurrent neural networks with context fusion for point cloud semantic segmentation," in *Proc. ECCV*, 2018, pp. 403–417.
[45] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proc. IEEE CVPR*, Jun. 2018, pp. 4558–4567.
[46] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*. [Online]. Available: https://arxiv.org/abs/1603.04467
[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980
[48] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3D object detection from RGB-D data," in *Proc. IEEE CVPR*, Jun. 2018, pp. 918–927.

Ying Li received the M.Sc. degree in remote sensing from Wuhan University, Wuhan, China, in 2017. She is currently pursuing the Ph.D. degree with the Mobile Sensing and Geodata Science Laboratory, Department of Geography and Environmental Management, University of Waterloo, Waterloo, ON, Canada.

Her research interests include autonomous driving, mobile laser scanning, intelligent processing of point clouds, geometric and semantic modeling, and augmented reality.



Lingfei Ma (S'18) received the B.Sc. and M.Sc. degrees in geomatics engineering from the University of Waterloo, Waterloo, ON, Canada, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree in photogrammetry and remote sensing with the Mobile Sensing and Geodata Science Laboratory, Department of Geography and Environmental Management.

His research interests include autonomous driving, mobile laser scanning, intelligent processing of point clouds, 3-D scene modeling, and machine learning.

Zilong Zhong (S'15) received the Ph.D. degree in systems design engineering, specialized in machine learning and intelligence from the University of Waterloo, Waterloo, ON, Canada, in 2019.

He is currently a Post-Doctoral Fellow with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His research interests include computer vision, deep learning, and graph models and their applications involving large-scale image analysis.

Dongpu Cao (M'08) received the Ph.D. degree from Concordia University, Montreal, QC, Canada, in 2008.

He is currently the Canada Research Chair of driver cognition and automated driving and an Associate Professor and the Director of the Waterloo Cognitive Autonomous Driving (CogDrive) Lab, University of Waterloo, Waterloo, ON, Canada. He has contributed more than 200 publications, two books, and one patent. His research focuses on driver cognition, automated driving, and cognitive autonomous driving.

Dr. Cao received the SAE Arch T. Colwell Merit Award in 2012 and three Best Paper Awards from the American Society of Mechanical Engineers (ASME) and IEEE conferences.

Jonathan Li (M'00–SM'11) received the Ph.D. degree in geomatics engineering from the University of Cape Town, Cape Town, South Africa, in 2000.

He is currently a Professor with the Departments of Geography and Environmental Management and System Design Engineering, University of Waterloo, Waterloo, ON, Canada. He has coauthored more than 400 publications, more than 200 of which were published in refereed journals. His research interests include information extraction from mobile LiDAR point clouds and from Earth observation images.

Dr. Li received the Outstanding Achievement Award from the Mobile Mapping Technology in 2019. He is an Associate Editor of the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, the International Society for Photogrammetry and Remote Sensing (ISPRS) *Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, and the *Canadian Journal of Remote Sensing*.