remote sensing

MDPI

Article

DGPolarNet: Dynamic Graph Convolution Network for LiDAR Point Cloud Semantic Segmentation on Polar BEV

Wei Song [1], Zhen Liu [1], Ying Guo [1,*], Su Sun [2], Guidong Zu [3] and Maozhen Li [4]

[1] School of Information Science and Technology, North China University of Technology, Beijing 100144, China
[2] Department of Computer and Information Technology, Purdue University, West Lafayette, IN 47907, USA
[3] COFCO Trading Agriculture & Big Data Solutions Co., Ltd., Dalian 116601, China
[4] Department of Electronic and Electrical Engineering, Brunel University London, Uxbridge UB8 3PH, UK
* Correspondence: guoying@ncut.edu.cn; Tel.: +86-132-6260-1918

Abstract: Semantic segmentation in LiDAR point clouds has become an important research topic for autonomous driving systems. This paper proposes a dynamic graph convolution neural network for LiDAR point cloud semantic segmentation using a polar bird's-eye view, referred to as DGPolarNet. LiDAR point clouds are converted to polar coordinates, which are rasterized into regular grids. The points mapped onto each grid distribute evenly to solve the problem of the sparse distribution and uneven density of LiDAR point clouds. In DGPolarNet, a dynamic feature extraction module is designed to generate edge features of perceptual points of interest sampled by the farthest point sampling and K-nearest neighbor methods. By embedding edge features with the original point cloud, local features are obtained and input into PointNet to quantize the points and predict semantic segmentation results. The system was tested on the Semantic KITTI dataset, and the segmentation accuracy reached 56.5%

Keywords: semantic segmentation; polar BEV; LiDAR point cloud; dynamic graph convolution network

Citation: Song, W.; Liu, Z.; Guo, Y.; Sun, S.; Zu, G.; Li, M. DGPolarNet: Dynamic Graph Convolution Network for LiDAR Point Cloud Semantic Segmentation on Polar BEV. *Remote Sens.* **2022**, *14*, 3825. https://doi.org/10.3390/rs14153825

Academic Editor: Henrique Lorenzo

Received: 20 June 2022
Accepted: 4 August 2022
Published: 8 August 2022

1. Introduction

LiDAR sensors are essential devices for environmental perception tasks in smart vehicles, as they can scan millions of 3D points for each frame [1–3]. In recent years, LiDAR-based semantic segmentation technology has achieved rapid development. However, LiDAR point clouds have the characteristics of irregular structure, uneven density, and sparse distribution, which are challenging problems for deep learning approaches.

Three-dimensional (3D) segmentation methods [4,5] based on machine learning, such as support vector machine (SVM), random forest, naïve Bayesian supervised learning, etc., usually utilize the geometrical or distribution features of point clouds to train models. The feature extraction [6,7] process of large-scale LiDAR point clouds is computationally intensive, which limits machine learning approaches for outdoor environment perception tasks. Meanwhile, because LiDAR point cloud density is high in the near field and loose in the far field, such methods have poor adaptability and expansion capabilities [8]. From the traditional machine learning method to various deep neural networks, the semantic segmentation approaches of projected views, points, voxels, and graphs have been widely researched. The multiview projection and voxel mapping methods lead to feature information loss. Meanwhile, due to the uneven point distribution density in far and near fields, the sampled unstable features cause low performance of the trained network model. PolarNet [9] has been proposed to rasterize the polar coordinates of LiDAR points into regular grids as input into a convolution neural network (CNN) model for semantic segmentation. Figure 1 compares the density distribution of a LiDAR point cloud frame in the Cartesian and polar BEV coordinate systems. The density distribution is more uniform under the polar BEV coordinate system. Although PolarNet solves the problem of the

uneven density of LiDAR point clouds, the applied feature extraction method by using max-pooling operations causes the information loss of detailed geometrical features.

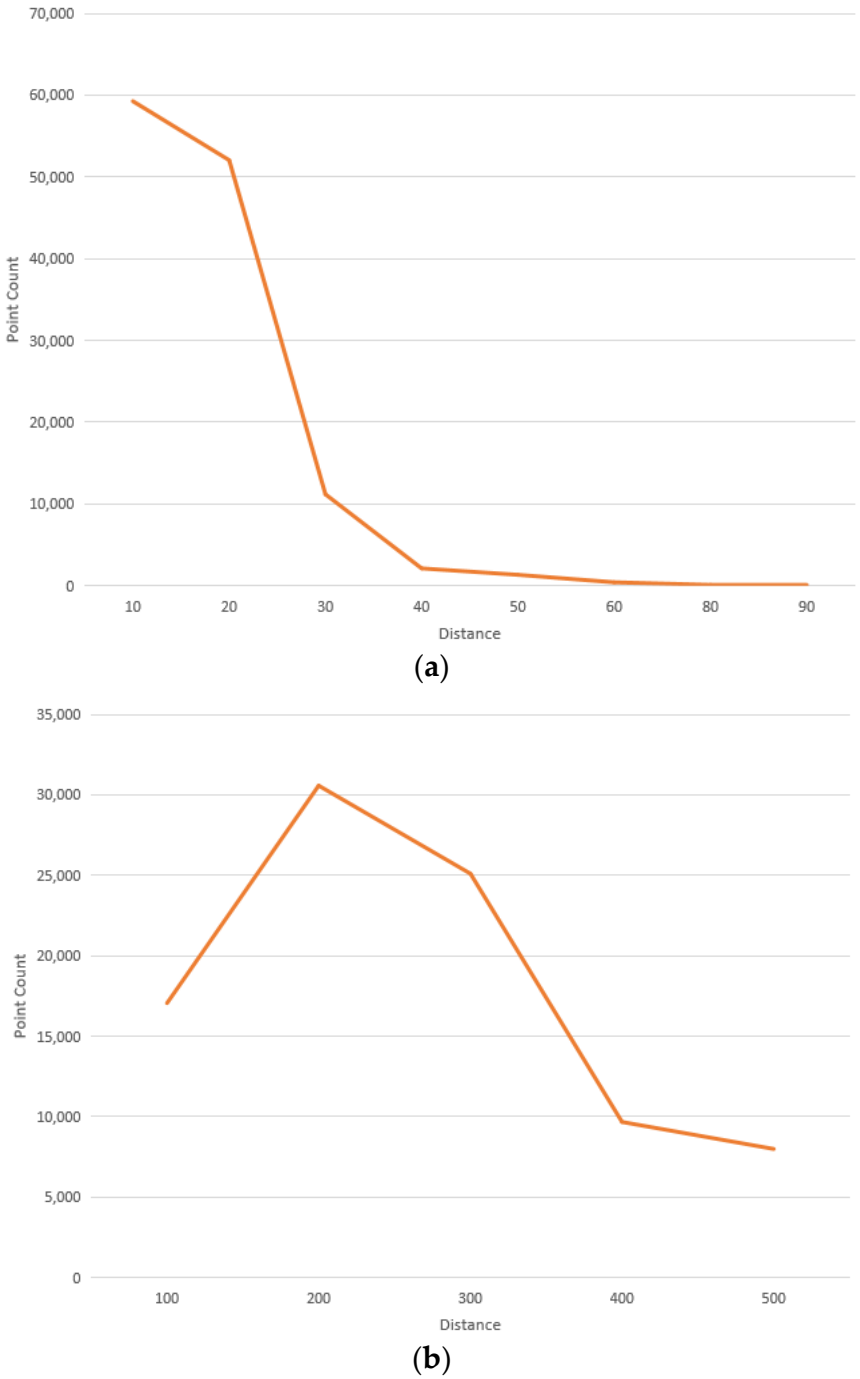


Figure 1. Density distributions of LiDAR point clouds in the Cartesian and polar BEV coordinate systems, where the *x*- and *y*-axes indicate distance and point count, respectively: (**a**) Cartesian coordinate system; (**b**) polar BEV coordinate system.

This paper proposes a dynamic graph convolution network for LiDAR point cloud semantic segmentation using a polar bird's-eye view, referred to as DGPolarNet, as shown in Figure 2. The proposed DGPolarNet input is the original point clouds, and the output is the semantic segmentation results. Firstly, the LiDAR point clouds are converted to polar coordinates, which are registered into regular grids to balance the input data to DGPolarNet. Then, a dynamic feature extraction module generates edge features based on perceptual

points of interest sampled by the farthest point sampling (FPS) and K-nearest neighbor (KNN) methods. Finally, the extracted edge features are combined with the original point cloud through skip connections to recover spatial information lost and enhance local features with describable semantic information. The extracted dynamical edge features are input into a convolutional neural network to provide discriminant features for the semantic segmentation network.

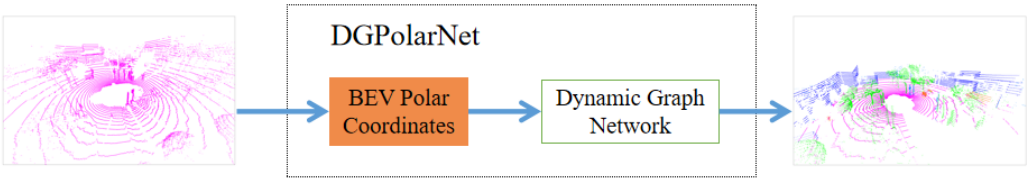


Figure 2. DGPolarNet framework for semantic segmentation from LiDAR point clouds.

The main contributions of this paper are as follows: (1) The semantic segmentation based on a polar bird's-eye view (BEV) solves the problems of the sparse distribution and uneven density of LiDAR point clouds. (2) The edge features generated from the points of interest sampled by FPS and KNN are more discriminants than the local features computed by KNN.

2. Related Works

In this section, we briefly survey the related works of 3D semantic segmentation, including neural network models to process converted multiviews, voxels, points, and graphs.

2.1. Multiview Projection Methods

To reuse deep neural network models for semantic segmentation in 2D images, 3D point clouds are projected onto 2D images or spherical images for pixel-wise semantic labeling. Lawin et al. [10], Boulch et al. [11], and Tatarchenko et al. [12] mapped 3D point clouds onto multiple 2D images from different viewports, which were input into deep learning networks, such as convolutional neural network (CNN), full convolutional network (FCN), and fully convolutional U-shaped network with skip connections, for pixel-wise semantic labeling. Using an inverse mapping procedure, semantic segmentation was implemented in 3D point clouds. Su et al. [13] proposed a sparse lattice network (SPLATNet) with a bilateral convolution layer. Original point clouds were mapped to a sparse lattice, in which valid grids were input into bilateral convolution layers for hierarchical and spatial features extraction. SPLATNet was a joint 2D–3D network with a series of 1×1 CONV layers for the fusion of the extracted features of 3D point clouds and multiview images. Wu et al. [14] proposed a semantic segmentation framework, SqueezeSeg. It utilized spherical projection to transform sparse and irregular 3D point clouds into dense 2D grids, which were input into a convolutional network, SqueezeNet, for feature extraction. The segmentation results were further optimized based on conditional random fields. Similarly, Milioto et al. [15] proposed RangeNet++ operating on the range views of spherical projection from LiDAR point clouds. To improve computation speed performance, they utilized a GPU parallel programming technique to speed up the postprocessing of KNN operations. However, such 3D semantic segmentation methods based on 2D projection were sensitive to viewport selection and had low accuracy performance when processing point clouds of sparse distribution.

Su et al. [16] proposed a multiview convolutional neural network (MVCNN) that captured 2D images from different viewports and aggregated them into 3D shape descriptors by convolutional and pooling operations. These descriptive features were fed into a semantic segmentation network. However, this method only allowed selecting fixed viewports and ignored a large amount of geometric space information, which was not suitable for large-scale complex scenes. To generate discriminative information, Feng et al. [17] proposed a group-view convolutional neural network (GVCNN) based on MVCNN. The

GVCNN used a fully convolutional network to extract view-level descriptors. In addition, grouping modules were used to learn association information and distinguish information between different viewports. Based on content-based discrimination, group-level descriptors were generated by dividing multiviews into several groups. The network applied a fully connected layer to complete the identification task. Classification performance was significantly improved by emphasizing discriminative information in the 3D shape descriptor, which was discovered at the group level. However, the GVCNN used maximum pooling for multiview clustering operations, which lost some useful information. Wang et al. [18] proposed a recurrent clustering and pooling convolutional neural network (RCPCNN). The RCP module was designed to generate dominant sets based on the similarity of multiple views. The RCPCNN grouped clusters of the dominant set, which were used for network updating by the pooling method. To improve the adaptability of the semantic segmentation network, the RCP module used the cyclic clustering and pooling module to generate feature vectors automatically. To fully exploit the correlation between multiple views, Ma et al. [19] combined the long short-term memory (LSTM) network with a CNN for 3D shape recognition. The 2D CNN network was used to extract the low-level features for each image of the view subsequences. The extracted features were input into the LSTM network as a time series. Through a sequence voting layer, the extracted features were aggregated into shape descriptors. This model fully utilized the advantages of CNN and LSTM, which effectively improved the discriminative ability of multiview descriptors.

2.2. Voxelization Methods

Maturana et al. [20] proposed VoxNet to convert unstructured point cloud data into grid data as the input of a 3D CNN. Point clouds were mapped to multiple 3D grids using an occupancy grid algorithm. The value of each grid cell was normalized and fed into the convolutional layers of the network to generate feature maps. A max-pooling method was performed on nonoverlapping blocks of voxels. Rethage et al. [21] proposed a fully convolutional point network (FCPN) for semantic voxel labeling. In low-level feature abstraction layers, the FCPN employed PointNet with a uniform sampling strategy to ensure the permutation invariance of local geometric feature extraction. The high-dimension feature extraction network formed an octree-like nonoverlapping feature space to reduce memory costs. To enhance the performance of convolutional networks on sparse 3D voxelized data, Graham et al. [22] designed submanifold sparse convolutional networks (SSCNs), which fixed the active sites of sparse convolutions to keep sparsity stable over multiple layers. To solve the submanifold dilation problem, the SSCNs only process active voxels. Moreover, strided operations incorporating pooling or strided convolution were introduced for data fusion in the hidden layers between disconnected components. Wu et al. [23] proposed 3DShapeNet, which represented 3D geometry features as a probability distribution of binary variables on 3D voxels. Binary tensors were fed into three layers of convolutional filters to extract features. Although these methods solved the problem of processing unstructured point clouds, the memory usage may increase cubically with the increase in resolution.

Riegler et al. [24] proposed OctNet with adaptive spatial partitioning capability that replaced fixed-resolution voxels with a flexible octree structure, effectively solving the problem of the high memory consumption of voxels. It divided the 3D space hierarchically into a set of unbalanced octree structures. In the octree structures, each leaf node stored a pooled feature representation. The 3D space was partitioned based on data density. Computational storage resources were dynamically pooled based on the input 3D structure, which greatly reduced memory consumption and improved computational speed. Wang et al. [25] proposed an O-CNN that implemented convolution operations only on sparse octants occupied by 3D surface boundaries. The O-CNN used the sparsity of the octree representation and the local orientation of the shape to reasonably allocate memory and improve computational efficiency. Xu et al. [26] proposed a learning-free 3D point cloud segmentation strategy based on the octree structure. The graph structure was gener-

ated based on local contextual information for point cloud voxelization and clustering of voxels. Perceptual grouping was utilized to segment 3D point clouds in a purely geometric way. This method had better classification results for complex scenes and objects with nonplanar surfaces. However, the traversing process of the octree structure caused high time complexity for generating a high-resolution octree.

To solve the problems of the high memory consumption and long training time of voxelized networks, Li et al. [27] proposed a field-probing neural network for 3D data (FPNN). The 3D space was represented by a 3D vector field as the input to the network. Instead of a convolutional layer in the CNN, a field-probing filter was used to extract features from the 3D vector field efficiently. This way, the computational complexity only depended on the number of field-probing filters and sampling points, and it was not affected by the resolution of the voxel map. Le et al. [28] proposed a point-grid hybrid model, PointGrid, which sampled a certain number of points in each grid cell with a simple point quantization strategy. From these sampled points, local ensemble features were extracted. Tchapmi et al. [29] proposed SEGCloud, which voxelized 3D point clouds and generated coarse downsampled voxel labels by using 3D-FCNN. The voxel labels were interpolated back to the 3D points by trilinear interpolation layers. The original 3D point features were combined with the resulting class labels after interpolation. By using fully connected conditional random fields, the final class labels were inferred to obtain a fine-grained semantic segmentation level. In SEGCloud, the shared computation using fully convolutional networks reduced a certain amount of computation consumption. Although the voxel-based semantic segmentation methods utilized lossy compression on sparse point clouds as preprocessing to reduce computational memory consumption, loss of geometric and topological information of point clouds was caused.

2.3. Point Methods

To avoid loss of original information by projection or voxelization, Qi et al. [30] proposed a point-wise semantic segmentation network, PointNet, that directly operated on unordered point clouds without sampling preprocessing. PointNet learned geometric features through shared multilayer perceptrons (MLPs), which were input into symmetrical max-pooling functions for global features extraction. However, PointNet based on point-wise MLP focused on the perception of global geometric information and ignored the topological structure of local features of points. To extend the PointNet applicability to complex scenes, Qi et al. [31] proposed a deep hierarchical feature learning network, PointNet++. PointNet++ exploited an individual PointNet to generate local features in multiscale set abstractions and utilized hierarchical concepts to extract global features.

Considering both orientation awareness and scale awareness, Jiang [32] proposed a PointSIFT module that supported various PointNet-based architectures. The module used orientation-encoding convolution to generate multiscale representation from eight orientations. A set abstraction module in the downsampling stage and a feature propagation module in the upsampling stage were combined to obtain the features. Li et al. [33] proposed SO-Net, which constructed a self-organizing map (SOM) to model the spatial distribution of point clouds. The SOM node features were aggregated into a global feature vector using average pooling. A parallel branching network with fully connected branches and convolutional branches was used to recover a single feature vector from the global features to represent the input point cloud.

Because LiDAR point clouds have an imbalanced distribution, several invalid grids have been registered in far fields for traditional grid rasterization methods [34,35]. BEV maps represent point cloud data from a top-down perspective without losing any scale and range information [36,37]. By projecting raw point clouds into a fixed-size polar BEV map, Zhang et al. [9] proposed a PolarNet that extracted the local features in polar grids and integrated them into a 2D CNN for semantic segmentation.

2.4. Graph Methods

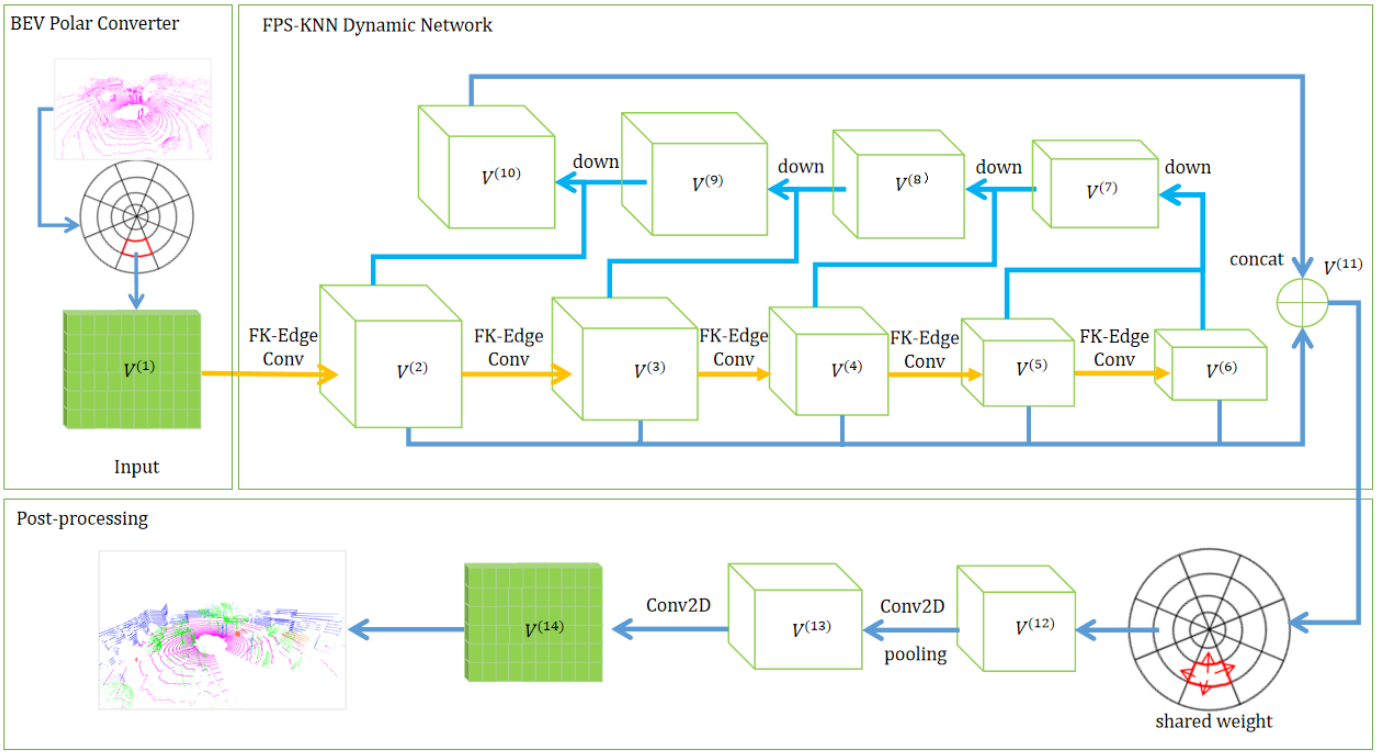
Currently, the graph neural network (GNN) research proposed by Scarselli et al. [38] has been widely studied for 2D and 3D semantic analysis tasks. Bruna et al. [39] applied convolutional neural networks in non-Euclidean domains modeled with graphs to perform local filtering in both the spatial and spectral domains. Simonovsky et al. [40] proposed an edge conditional convolution network on a local graph neighborhood. By integrating edge labels and an asymmetric edge function, the relationship between local points was established. Landrieu et al. [41] introduced an attribute-directed graph hyper point graph with edge features. Gated graph neural networks and edge conditional convolution were utilized to obtain contextual information for large-scale point cloud segmentation. Landrieu et al. [42] proposed a 3D point cloud oversegmentation strategy to improve segmentation accuracy. Jiang et al. [43] aggregated point features into edge branches in a hierarchical manner to enhance the description of local features. Kipf et al. [44] proposed a semi-supervised classification network, the graph convolutional neural networks (GCN). To extend the applicability of the GCN, Te et al. [45] proposed a regularized graph convolutional neural network (RGCNN), which mainly consisted of graph construction, graph convolution, and feature filtering layers. In each convolution layer, the graph Laplacian matrix was updated to improve the flexibility to adapt to dynamic graphs. The Chebyshev polynomial was also used to reduce computational complexity. Experiments indicated that the RGCNN had good performances in both point cloud classification and segmentation tasks of different densities. Wang et al. [46] developed a dynamic graph convolutional neural network (DGCNN). To obtain the local features of point clouds, the DGCNN used the EdgeConv operation to extract the features of centroids in a local neighborhood map. EdgeConv only considered the point coordinates and the distances of the neighboring points. Because the vector directions between neighboring points were ignored, some local geometric information was eventually lost in this method.

To enhance local feature representation, this paper proposes DGPolarNet, an inheritance derived from DGCNN, PolarNet, and PointNet++. The local discriminate edge features generated from the points of interest sampled by FPS and KNN discriminate more than the local features computed by KNN. The extracted local features and the original point cloud data are aggregated for semantic segmentation to prevent feature loss during the convolution and pooling operations.

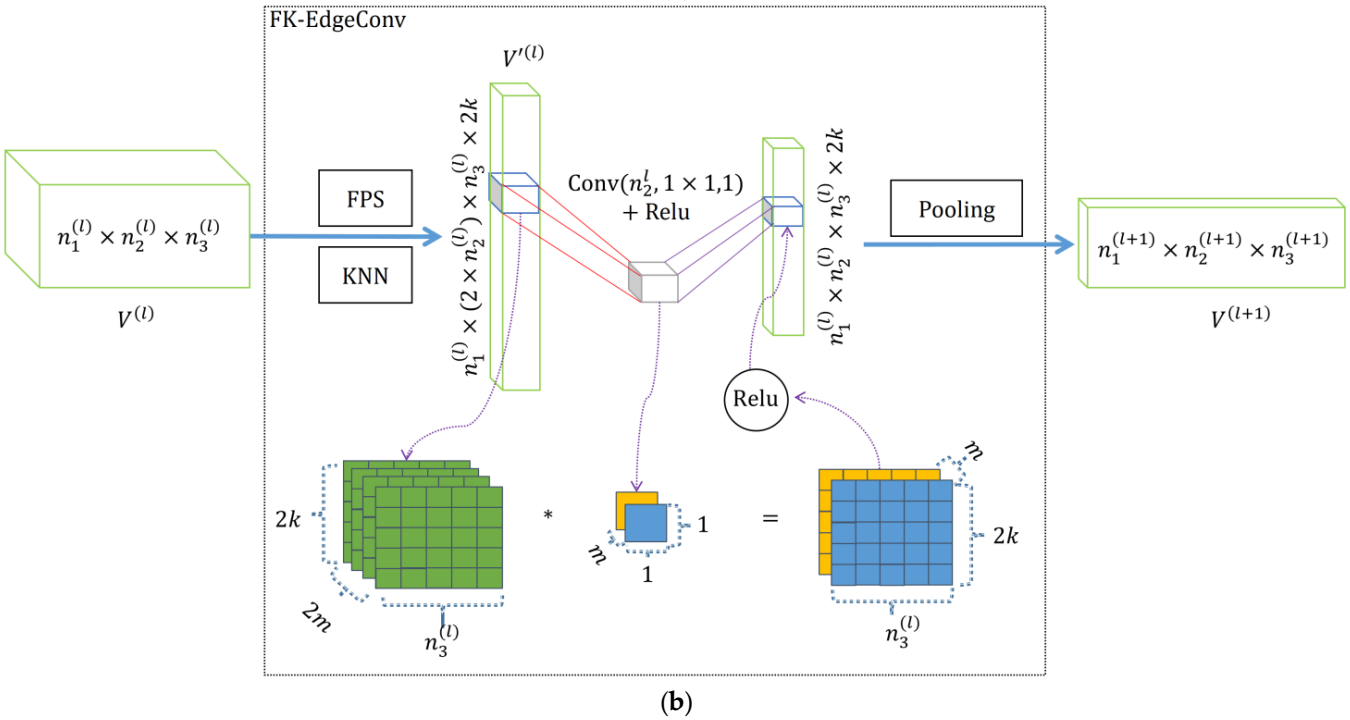
3. DGPolarNet for LiDAR Point Cloud Semantic Segmentation

Through a comprehensive analysis of global and local features, this paper proposes DGPolarNet, a dynamic graph convolution network with FPS and KNN for LiDAR point cloud semantic segmentation based on a polar BEV, as shown in Figure 3. DGPolarNet mainly consists of an FPS-KNN dynamic network and shared MLP postprocessing.

The FPS-KNN dynamic network is developed to convert unstructured raw LiDAR point clouds into high-dimensional features for semantic segmentation. The input of the network is the original LiDAR point clouds, and the output is the semantic labels of all points. The original point cloud is converted to polar representation by a polar BEV converter unit. The FPS-KNN dynamic network module implements edge feature extraction and feature fusion to obtain the aggregated features. The postprocessing module integrates all the aggregation features of the whole polar BEV map for semantic analysis.

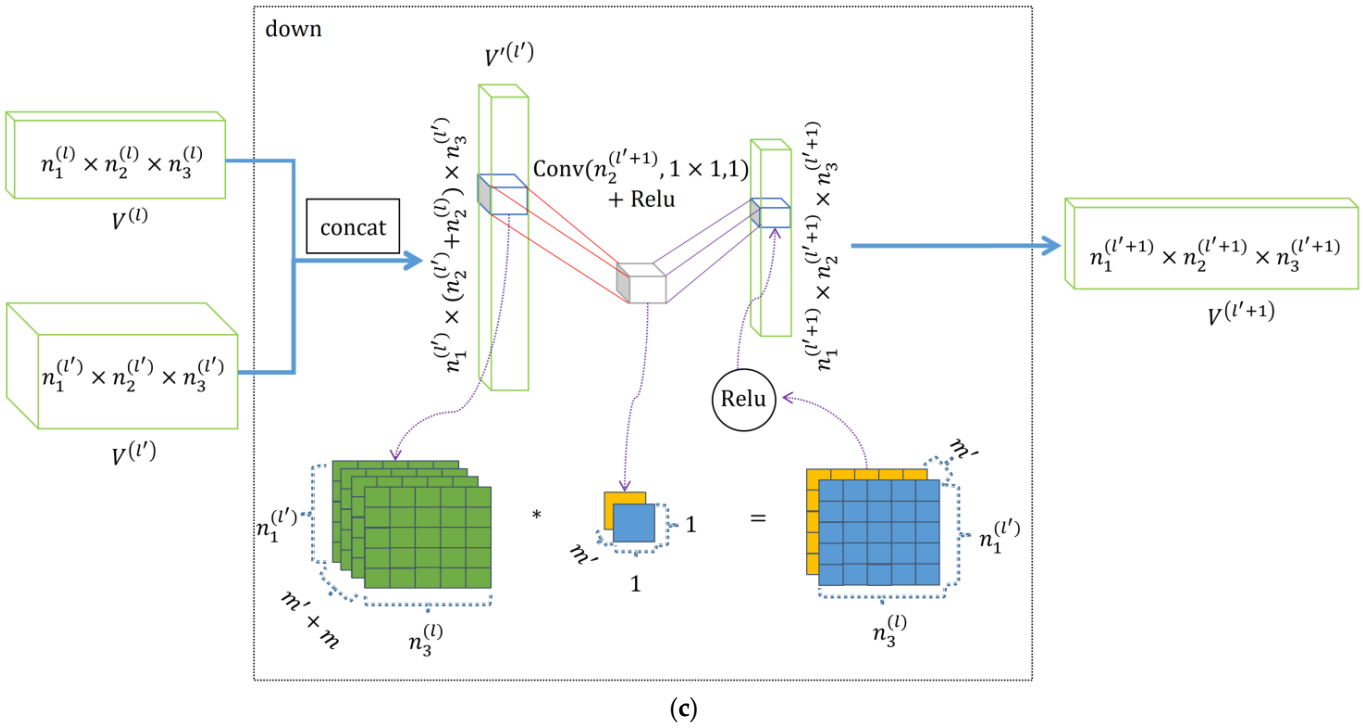


(**a**)



(**b**)

Figure 3. *Cont.*



(**c**)

Figure 3. Proposed semantic segmentation method: (**a**) DGPolarNet framework; (**b**) FPS-KNN dynamic network for edge feature extraction of polar BEV blocks; (**c**) downsampling process for high-dimensional edge feature generation.

3.1. BEV Polar Converter

To solve the invalid grid waste problem of processing the uneven distribution of LiDAR scanning, the polar BEV coordinate system is utilized to register 3D point clouds into regular grids. Using Equation (1), the 3D point (x, y, z, t) is converted into the polar coordinate (r, θ, t), where (r, θ) is the polar coordinate defined by Equations (1) and (2), and t is the intensity value of the laser reflection.

$$r = \sqrt{x^2 + y^2 + z^2} \tag{1}$$

$$\theta = \arcsin\frac{\mathrm{y}}{\sqrt{x^2 + y^2 + z^2}} \tag{2}$$

The points in the polar BEV grid defined as $p_i(r_i, \theta_i, t_i) \in P$ are rasterized into a 3D array $V^{(1)}$ of size $\left(n_1^1 \times n_2^1 \times n_3^1\right)$, which is then input into the FPS-KNN dynamic network as the first layer of the backbone network. For the first layer $V^{(1)}$, n_1^1 is the batch size, and n_3^1 is the number of points in each batch. Each point has three attributes $\{r, \theta, t\}$; thus, $n_2^1 = 3$.

3.2. FPS-KNN Dynamic Network

The FK-EdgeConv (FPS-KNN EdgeConv) method is developed by integrating FPS and KNN algorithms to extract the comprehensive edge features of the nearest and farthest vertices, as shown in Figure 4.

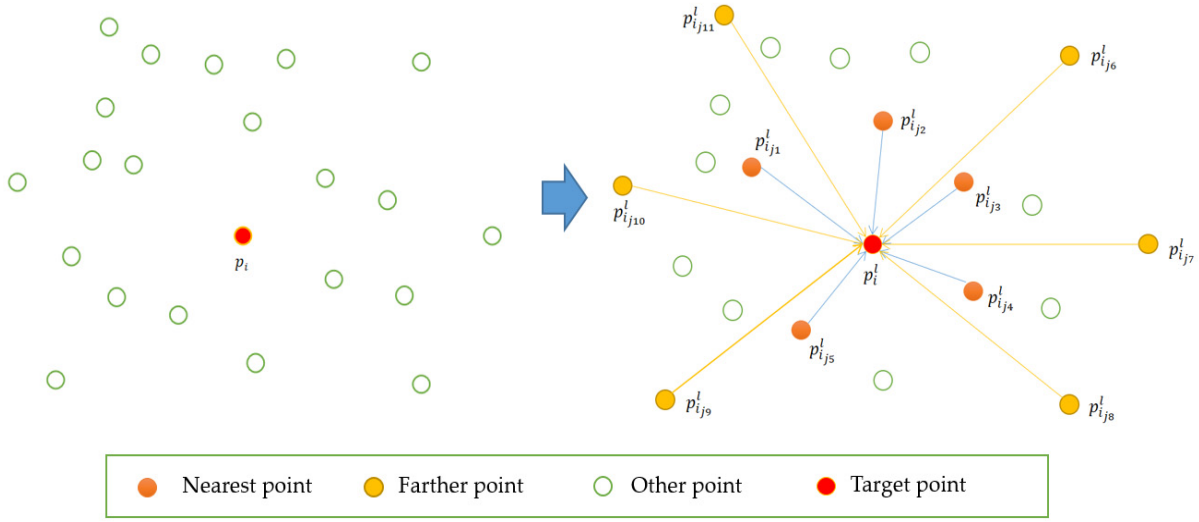


Figure 4. Example of edge feature generation by computing the vectors between the vertex $p_i^{(l)}$ from its corresponding nearest and farthest neighbors sampled by FPS and KNN.

The FPS-KNN dynamic network constructs a directed graph for each layer using the FK-EdgeConv method. For the l-th network layer, the dynamic graph $G^{(l)}$ is defined as Equation (3), where the datasets $V^{(l)}$ and $E^{(l)}$ with a dimension of $(n_1^{(l)} \times n_2^{(l)} \times n_3^{(l)})$ represent the set of vertices and edges, respectively.

$$G^{(l)} = \left(V^{(l)}, E^{(l)}\right) \tag{3}$$

$$V^{(l)} = \left\{ p_i^{(l)} \middle| i = 1, 2, \ldots, n^{(l)} \right\} \tag{4}$$

$$E^{(l)} = \left\{ \varepsilon_i^{(l)} = (\varepsilon_{i_1}^{(l)}, \varepsilon_{i_2}^{(l)}, \ldots, \varepsilon_{i_{2k^{(l)}}}^{(l)}) \middle| i = 1, 2, \ldots, n^{(l)} \right\} \tag{5}$$

$$\varepsilon_{i_j}^{(l)} = p_{i_j}^{(l)} - p_i^{(l)} \tag{6}$$

To obtain more effective semantic features, FPS and KNN are utilized to generate directed graphs from the LiDAR point clouds instead of the fully connected edges, which suffer from high memory consumption. The FPS and KNN operations sample the $k^{(l)}$ farthest and $k^{(l)}$ nearest neighbors, respectively, from the vertices set $V^{(l)}$. Thus, the edge set $E^{(l)}$ has $2 \times k^{(l)}$ directed edge elements, which are calculated based on the target point $p_i^{(l)}$ and the neighbor points using Equation (6).

Then, the vertices in $V^{(l)}$ and the edges in $E^{(l)}$ are input into the Conv2D and pooling operations to generate the output dataset $V^{(l+1)}$ with a dimension of $(n_1^{(l+1)} \times n_2^{(l+1)} \times n_3^{(l+1)})$, in which $n_1^{(l+1)} = n_1^{(l)}$, $n_2^{(l+1)} = n_2^{(l)}$, and $n_3^{(l+1)} = n_3^{(l)}$. The edge feature computation as the Conv2D is defined as $h\left(p_i, \varepsilon_{i_j}\right)$. We utilize max-pooling and min-pooling operations to extract local features from the sampled farthest and nearest vertices, respectively. Accordingly, the output $p_i^{(l+1)} \in V^{(l+1)}$ of the FK-EdgeConv operation is denoted as Equation (7). The dataset $V^{(l+1)}$ is also the input of the (l+1)-th layer processed by the following FK-EdgeConv operation. In particular, only the directed graph of the first layer of the FPS-KNN dynamic

network is built based on the points in the polar BEV coordinate system. The following layers are constructed using the features extracted from the previous layer.

$$p_i^{(l+1)} = \begin{cases} max\{h\left(p_i^{(l)}, \varepsilon_{i_1}^{(l)}\right), h\left(p_i^{(l)}, \varepsilon_{i_2}^{(l)}\right), \ldots\ldots, h\left(p_i^{(l)}, \varepsilon_{i_k}^{(l)}\right) | i = 1, 2, \ldots, k\} \\ min\{h\left(p_i^{(l)}, \varepsilon_{i_1}^{(l)}\right), h\left(p_i^{(l)}, \varepsilon_{i_2}^{(l)}\right), \ldots\ldots, \left(p_i^{(l)}, \varepsilon_{i_k}^{(l)}\right) | i = k+1, k+2, \ldots, 2k\} \end{cases} \quad (7)$$

The FK-EdgeConv unit, as shown in Figure 3b, is developed to compute local features of the array $V^{(l)} = \{v_1^{(l)}, v_2^{(l)}, v_3^{(l)}\}$ of size $(n_1^{(l)} \times n_2^{(l)} \times n_3^{(l)})$ for the l-th layer. The FPS and KNN algorithms are implemented on $V^{(l)}$ and output the feature array $V'^{(l)} = \{v_1^{(l)}, \overline{v}_2^{(l)}, v_3^{(l)}, v_4^{(l)}\}$ of size $(n_1^{(l)} \times (2 \times n_2^{(l)}) \times n_3^{(l)} \times 2k)$, which contains both the point and edge features of the k nearest points and the k farthest points. By using Conv+Relu and pooling operations on each batch of $V'^{(l)}$, the local feature array $V^{(l+1)}$ of size $\left(n_1^{l+1} \times n_2^{l+1} \times n_3^{l+1}\right)$ is generated as the input of the following layer. In each layer, FK-EdgeConv is utilized to calculate the dynamic feature graph model as local semantic features, which are further aggregated for semantic feature enhancement. In our practice, we implement five FK-EdgeConv operations and four down operations accordingly.

After extracting the graph features of multiple layers, the down unit, as described in Figure 3c, is implemented to fuse the extract features of the layers l and l'. The inputs of the down unit consist of two feature sets of different sizes. The feature arrays $V^{(l)}$ and $V^{(l')}$ are reshaped and concatenated into the aggregated feature array $V\prime^{(l')} = \{v_1^{(l')}, v_2^{(l')} + v_2^{(l)}, v_3^{(l')}\}$ of size $(n_1^{(l')} \times (n_2^{(l')} + n_2^{(l)}) \times n_3^{(l')})$. Using the Conv+Relu operation on each batch of $V\prime^{(l')}$, the higher-dimensional feature array $V^{(l'+1)}$ is generated as the input of the following layer.

By using a skip architecture, the local features of FK-EdgeConv operations and down processes are joined as aggregated features, which are input into the postprocessing for global semantic segmentation. Similar to the skip architecture, the features generated by the FPS-KNN EdgeConv and down processes are reshaped by cropping and scaling operations to merge into the aggregated features in the concatenating procedure.

3.3. Postprocessing

The aggregation features generated by the FPS-KNN dynamic network are mapped back to their corresponding polar BEV grid as the input of postprocessing. The shared MLP is utilized to convolute the semantic segmentation prediction. In the l-th layer, the feature set v^l is computed via Equation (8), where w_{pqt}^{lm} is the learnable parameter for the element (p, q, t) in layer m of the MLP.

$$v_{ijk}^l = relu\left(b^l + \sum_m \sum_{p=0}^{p_l-1} \sum_{q=0}^{q_l-1} \sum_{t=0}^{R_l-1} w_{pqt}^{lm} v_{(r+i)(\theta+j)(t+c)}^{(l-1)m}\right) \quad (8)$$

4. Experiments and Analysis

The proposed model was tested on the SemanticKITTI [47] datasets. Compared with other typical semantic segmentation networks, the accuracy performances of the DGPolarNet model with several critical parameters are analyzed and discussed in this section.

4.1. Datasets

SemanticKITTI is a dataset of LiDAR point clouds collected by Velodyne Lidar HDL-64E and annotated with point-level semantic labels. It consists of a total of 43,551 frames from 22 sequences collected from inner-city traffic, including 23,201 for training and the rest for testing. Each frame has around 104,452 points on average. There are a total number of 19 typical types of objects, including ground-related, structures, vehicles, nature, human, object, and outlier classes. The dataset is unbalanced in point counts for different objects. For example, there is a small number of motorcyclist objects in most scenes, and only a few

points are labeled for the motorcyclist class. In our experiment, we used one sequence for validation and nine sequences for training.

4.2. Semantic Segmentation Performance

Experiments in this section were conducted using an Intel(R) Xeon(R) Silver 4110 CPU @ 2.10 GHz 2.10 GHz dual processor, an NVIDIA Quadro RTX 6000 graphics card, and 64 GB RAM. In our experiments, the points in the range of $-50\,\text{m} < x < 50\,\text{m}$, $-50\,\text{m} < y < 50\,\text{m}$, and $-3\,\text{m} < z < 1.5\,\text{m}$ were mapped to the polar BEV coordinate system, which were then rasterized into the polar BEV grids with a resolution of $(480 \times 360 \times 32)$. After analyzing the point distribution of the SemanticKITTI dataset, we specified the k value of each layer as equal to 20. In our experiment, we specified $v_1^1 = 1$, $v_2^1 = 3$, and $v_3^1 = 1{,}843{,}200$. Our DGPolarNet model had 14 layers, among which the 1st layer was the converted polar BEV grid data, the 2nd to 11th layers were FPS-KNN dynamic networks, and the 12th to 14th layers were for postprocessing. $V^{(11)}$ represented the aggregated features of the 11th layer, and $V^{(14)}$ represented the final semantic score of the 14th layer. The softmax function was utilized as the loss function. Table 1 illustrates the data dimension for each layer. In the 14th layer, there were 19 segmentation scores for 19 classes in SemanticKITTI accordingly.

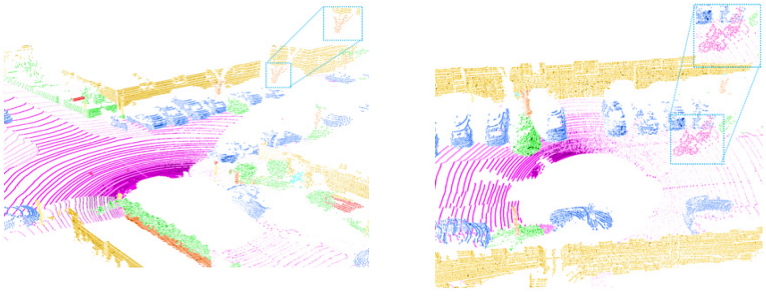
Table 1. Data dimensions for each layer.

l	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$n_1^{(l)}$	1	1	1	1	1	1	1	1	1	1	1	1	1	1
$n_2^{(l)}$	3	3	3	3	3	3	3	3	3	3	3	18	1024	19
$n_3^{(l)}$	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m	1.84 m

Figure 5 shows some samples of the semantic segmentation results using the proposed DGPolarNet method. To evaluate the semantic segmentation performance, the mean intersection-over-union (mIoU) is applied (Equation (9)), where the variables TP_c, FP_c, and FN_c are the number of true-positive, false-positive, and false-negative predictions for class c, respectively, and C is the number of classes.

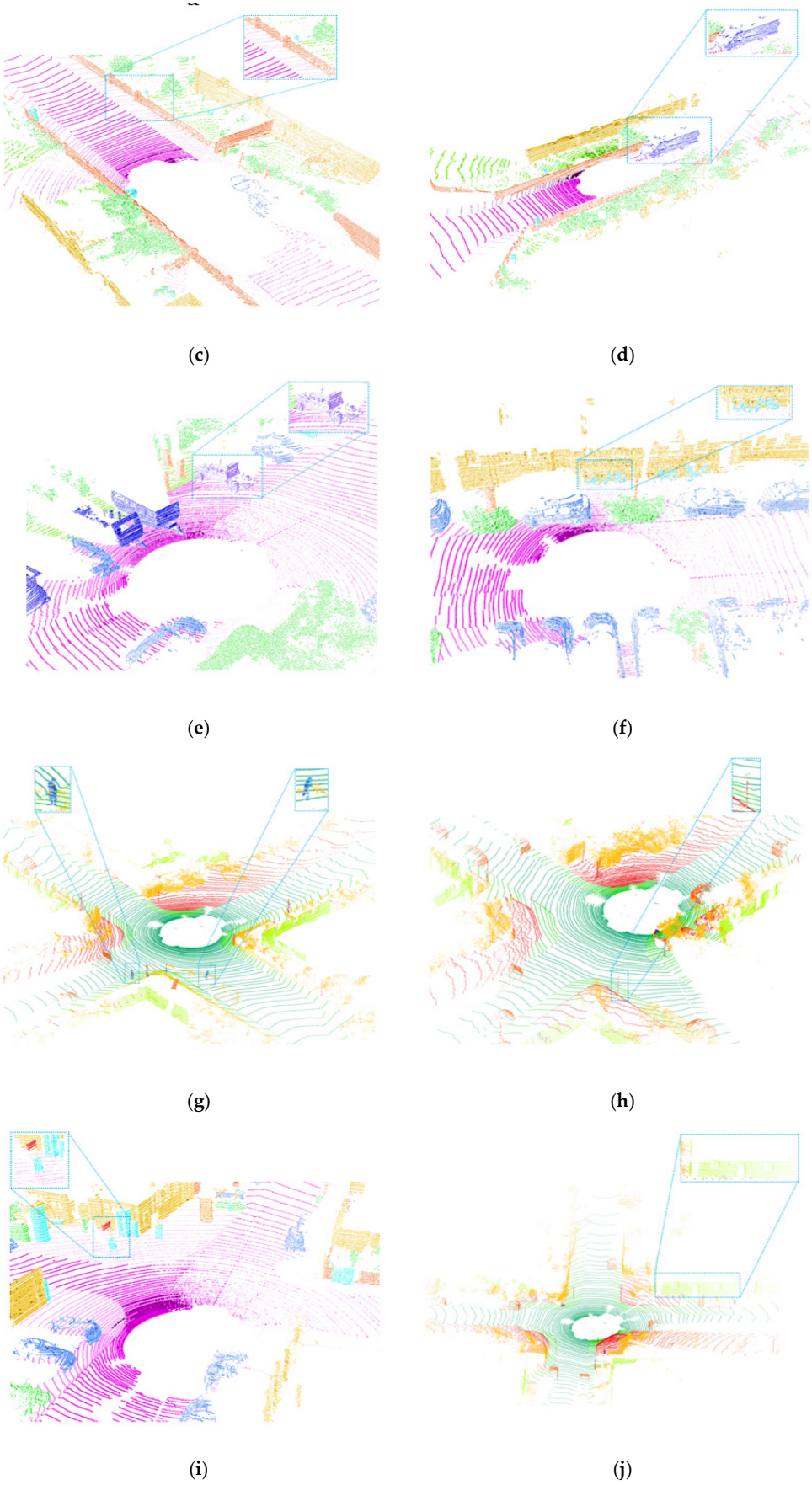
$$\text{mIoU} = \frac{1}{C}\sum_{c=1}^{C}\frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c + \text{FN}_c} \tag{9}$$

Table 2 shows the segmentation mIoU performances on all the object classes of the SemanticKITTI compared with the state-of-the-art methods. Our mIoU achieved 56.5% on average. Using the proposed DGPolarNet, the average segmentation IoUs of ground-related regions, buildings, vehicles, nature regions, humans, and other objects were 86.4%, 90.1%, 45.20%, 81.35%, 43.83%, and 52.40%, respectively. Our method had good performance for motorcycles, trucks, bicyclists, roads, sidewalks, buildings, vegetation, and pole objects. However, its IoU was low for other-ground, motorcyclist, and bicycle objects, because the extracted features of these objects were not discriminative.



(**a**) (**b**)

Figure 5. *Cont.*



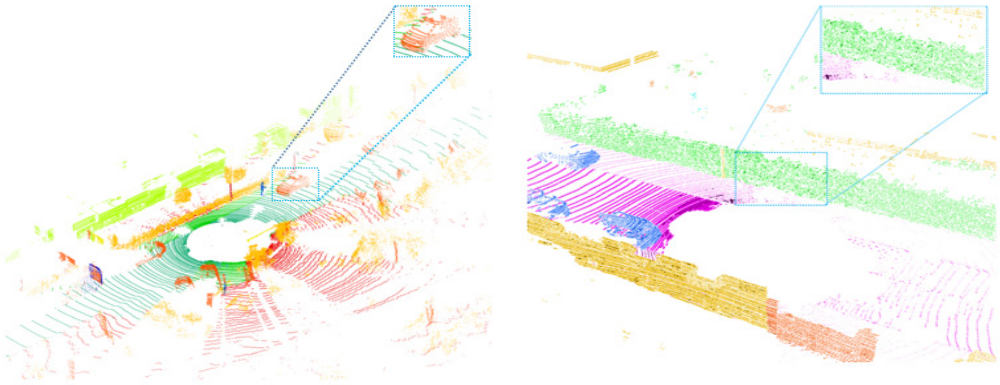
(**c**) (**d**)

(**e**) (**f**)

(**g**) (**h**)

(**i**) (**j**)

Figure 5. *Cont.*



(**k**) (**l**)

Figure 5. Semantic segmentation results by using the proposed DGPolarNet method on the SemanticKITTI: (**a**) Trunk; (**b**) Bicyclelist; (**c**) Fence; (**d**) Bus; (**e**) Truck; (**f**) Bicycle; (**g**) Person; (**h**) Pole; (**i**) Traffic-sign; (**j**) Building; (**k**) Car; (**l**) Vegetation.

Table 2. The IoU performances on SemanticKITTI compared with the typical semantic segmentation methods.

Model	MIoU	Per Class IoU								
		Ground-Related				Structures		Vehicle		
		Road	Sidewalk	Parking	Other-Ground	Building	Car	Truck	Bicycle	Motorcycle
PointNet [30]	14.60%	61.6%	35.7%	15.8%	1.4%	41.4%	46.3%	0.1%	1.3%	0.3%
PointNet++ [31]	20.10%	72.0%	41.8%	18.7%	5.6%	62.3%	53.7%	0.9%	1.9%	0.2%
SPG [41]	17.40%	45.0%	28.5%	0.6%	0.6%	64.3%	49.3%	0.1%	0.2%	0.2%
Squeezeseg [14]	29.50%	85.4%	54.3%	26.9%	4.5%	57.4%	68.8%	3.3%	16.0%	4.1%
TangentConv [12]	35.90%	82.9%	61.7%	15.2%	9.0%	82.8%	86.8%	11.6%	1.3%	12.7%
Squeezesegv2 [48]	39.70%	88.6%	67.6%	45.8%	17.7%	73.7%	81.8%	13.4%	18.5%	17.9%
DarkNet53 [47]	49.90%	91.8%	74.6%	64.8%	**27.9%**	84.1%	86.4%	25.5%	24.5%	32.7%
RangeNet++ [15]	52.20%	91.8%	75.2%	**65.0%**	27.8%	87.4%	91.4%	25.7%	25.7%	34.4%
RandLA [49]	53.90%	90.7%	73.7%	60.3%	20.4%	86.9%	94.2%	**40.1%**	26.0%	25.8%
PolarNet [9]	54.30%	90.8%	74.4%	61.7%	21.7%	90.0%	**93.8%**	22.9%	**40.3%**	30.1%
DGPolarNet	**56.50%**	**93.4%**	**79.4%**	58.4%	20.0%	**90.1%**	92.6%	51.5%	18.5%	**38.9%**

Model	Per Class IoU									
	Vehicle	Nature		Human				Object		
	Other-Vehicle	Vegetation	Trunk	Terrain	Person	Bicyclist	Motorcyclist	Fence	Pole	Traffic-Sign
PointNet [30]	0.8%	31.0%	4.6%	17.6%	0.2%	0.2%	0.0%	12.9%	2.4%	3.7%
PointNet++ [31]	0.2%	46.5%	13.8%	30.0%	0.9%	1.0%	0.0%	16.9%	6.0%	8.9%
SPG [41]	0.8%	0.8%	27.2%	24.6%	0.3%	2.7%	0.1%	20.8%	15.9%	0.8%
Squeezeseg [14]	3.6%	60.0%	24.3%	53.7%	12.9%	13.1%	0.9%	29.0%	24.5%	24.5%
TangentConv [12]	10.2%	75.5%	42.5%	55.5%	17.1%	20.2%	0.5%	44.2%	22.2%	22.2%
Squeezesegv2 [48]	14.0%	71.8%	35.8%	60.2%	20.1%	25.1%	3.9%	41.1%	36.3%	36.3%
DarkNet53 [47]	22.6%	78.3%	50.1%	64.0%	36.2%	33.6%	4.7%	55.0%	52.2%	52.2%
RangeNet++ [15]	23.0%	80.5%	55.1%	64.6%	38.3%	38.8%	4.8%	58.6%	55.9%	55.9%
RandLA [49]	**39.9%**	81.4%	**66.8%**	49.2%	49.2%	48.2%	7.2%	56.3%	38.1%	38.1%
PolarNet [9]	28.5%	84.0%	65.5%	67.8%	43.2%	40.2%	5.6%	**61.3%**	51.8%	**57.5%**
DGPolarNet	21.0%	**86.8%**	57.7%	**75.9%**	**55.1%**	**66.8%**	**9.6%**	55.2%	**62.6%**	39.4%

PointNet [30] extracted the global features from all the points directly and lacked the correlation among the local features. Meanwhile, PointNet implemented the semantic segmentation only using the 3D point coordinates without the intensity information. The laser reflection intensities of different materials were distinguished from each other. Without the intensity information, the connected objects of different types were easily detected as one object. Thus, we introduced the intensity data as one input of the DGPolarNet to enhance the discriminative local features of the dynamic graph. Compared with PointNet, the mIoU performance of our model was improved by 41.9%. For the ground-related, road,

sidewalk, parking, and other-ground regions, our method increased by 31.8%, 43.7%, 42.6%, and 18.6%, respectively.

RangeNet++ [15] projected the original point clouds onto the 2D range view, which caused spatial structure information loss and rasterization errors. In particular. when processing vehicle and human objects, the mIoU was only 27.3% and 14.2%, respectively. We used the polar BEV converter to solve the problem of uneven distribution of point clouds and applied both FPS and KNN to preserve the local geometrical features. Thus, the mIoU of our model was improved by around 4.3% compared with RangeNet++ and much improved for the vehicle and human objects.

Although PolarNet [9] utilized the polar BEV system to balance the input distribution, the extracted feature for each grid cell was insufficient by a learnable simplified PointNet with a max-pooling operation. To retain geometrical features, we constructed the dynamic graph for each BEV grid. Meanwhile, the extracted high-level semantic features were enhanced by the skip architecture of all the intermediate layers. Compared with the PolarNet network, the mIoU of our model is improved by 2.2%. For objects of complex shapes, such as vehicles and humans, our method improved by 14.5% on average.

Instead of only using KNN, we sampled both farthest and nearest neighbors by integrating FPS and KNN algorithms, which reduced the discriminative feature loss in the local feature encoding process. We conducted the comparison experiments using the KNN method and FPS-KNN method under different k values in dynamic graph construction, as shown in Table 3. When the k value was specified as 20, the models achieved the best performance. When it increased higher, the performance of the two models degraded on the contrary. Because the FPS-KNN method constructed feature maps obtaining both the internal geometrical structures and external contours of objects, the encoded features of the dynamic graphs were more describable than only using KNN.

Table 3. Performances for the feature encoding models with different k values.

k Value	mIoU with FPS + KNN	MIoU with KNN
3	52.4	45.6
5	54.7	49.7
10	55.5	53.3
20	56.5	54.5
25	54.3	52.2

Table 4 analyzes the DGPolarNet performances through true-positive (TP), false-negative (FN), and true-negative (TN) samples of the semantic segmentation results. Accordingly, the precision (P), recall (R), and F1 scores were calculated by using Equation (10) to evaluate the semantic segmentation performances.

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{10}$$

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{11}$$

$$\mathrm{F1} = 2 \times \frac{\mathrm{P} \times \mathrm{R}}{\mathrm{P} + \mathrm{R}} \tag{12}$$

Table 4. Semantic segmentation performance of DGPolarNet.

Object Class	P	R	F1
Road	0.96	0.97	0.96
Sidewalk	0.82	0.93	0.88
Parking	0.69	0.59	0.64
Other-ground	0.23	0.20	0.22
Building	0.90	0.90	0.90
Car	0.97	0.96	0.96
Truck	0.57	0.67	0.62
Bicycle	0.21	0.25	0.23
Motorcycle	0.43	0.51	0.47
Other vehicle	0.23	0.27	0.25
Vegetation	0.70	0.86	0.78
Trunk	0.63	0.64	0.64
Terrain	0.61	0.75	0.67
Person	0.44	0.54	0.48
Bicyclist	0.67	0.86	0.75
Motorcyclist	0.11	0.13	0.12
Fence	0.68	0.60	0.64
Pole	0.72	0.83	0.78
Traffic-sign	0.79	0.46	0.57

The values of P, R, and F1 values in Table 4 indicated that the proposed DGPolarNet has lower segmentation accuracies for other-ground, bicycle, and motorcyclist objects than the other objects. Because the point distribution of other-ground was similar to that of road and sidewalk objects, and motorcyclist objects were also similar to person objects, the segmentation for such objects did not perform well. For bicycle objects, there were a small number of points scanned on the sample surface, which caused insufficient training of the network model.

5. Conclusions

This paper proposes DGPolarNet, an efficient approach for semantic segmentation in LiDAR point clouds. The polar BEV converter is utilized to rasterize the LiDAR points into regular polar grids of even point distribution. An FPS-KNN dynamic network is developed to construct dynamic directed graphs and extract the local features of each BEV grid. Employing skip connection, the graph features of each layer are aggregated into high-dimensional features. All the aggregated features of each BEV grid are then integrated into a shared MLP for semantic segmentation. We validate the proposed DGPolarNet on the SemanticKITTI dataset, which is more efficient than previous methods.

Author Contributions: W.S. and Y.G. contributed to the conception of the study, Z.L. performed the data analyses and wrote the manuscript, S.S. and G.Z. contributed significantly to the experiment and analysis, and M.L. helped perform the analysis with constructive discussions. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Education and Teaching Reform Project of North China University of Technology, Beijing Urban Governance Research Base, the Ministry of Science (MSIT, ICT), Korea, under the High-Potential Individuals Global Training Program (2020-0-01576) supervised by the Institute for Information and Communications Technology Planning and Evaluation (IITP), the Great Wall Scholar Program (CIT&TCD20190304), and the National Natural Science Foundation of China (No. 61503005). (Corresponding authors: Ying Guo).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ballouch, Z.; Hajji, R.; Poux, F.; Kharroubi, A.; Billen, R. A Prior Level Fusion Approach for the Semantic Segmentation of 3D Point Clouds Using Deep Learning. *Remote Sens.* **2022**, *14*, 3415. [CrossRef]
2. Wei, M.; Zhu, M.; Zhang, Y.; Sun, J.; Wang, J. Cyclic Global Guiding Network for Point Cloud Completion. *Remote Sens.* **2022**, *14*, 3316. [CrossRef]
3. Song, W.; Li, D.; Sun, S.; Zhang, L.; Xin, Y.; Sung, Y.; Choi, R. 2D&3DHNet for 3D Object Classification in LiDAR Point Cloud. *Remote Sens.* **2022**, *14*, 3146. [CrossRef]
4. Decker, K.T.; Borghetti, B.J. Composite Style Pixel and Point Convolution-Based Deep Fusion Neural Network Architecture for the Semantic Segmentation of Hyperspectral and Lidar Data. *Remote Sens.* **2022**, *14*, 2113. [CrossRef]
5. Liu, R.; Tao, F.; Liu, X.; Na, J.; Leng, H.; Wu, J.; Zhou, T. RAANet: A Residual ASPP with Attention Framework for Semantic Segmentation of High-Resolution Remote Sensing Images. *Remote Sens.* **2022**, *14*, 3109. [CrossRef]
6. Xu, T.; Gao, X.; Yang, Y.; Xu, L.; Xu, J.; Wang, Y. Construction of a Semantic Segmentation Network for the Overhead Catenary System Point Cloud Based on Multi-Scale Feature Fusion. *Remote Sens.* **2022**, *14*, 2768. [CrossRef]
7. Shuang, F.; Li, P.; Li, Y.; Zhang, Z.; Li, X. MSIDA-Net: Point Cloud Semantic Segmentation via Multi-Spatial Information and Dual Adaptive Blocks. *Remote Sens.* **2022**, *14*, 2187. [CrossRef]
8. Eeinmann, M.; Jutzi, B.; Hinz, S.; Mallet, C. Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. *ISPRS J. Photogramm. Remote Sens.* **2015**, *105*, 286–304. [CrossRef]
9. Zhang, Y.; Zhou, Z.; David, P.; Yue, X.; Xi, Z.; Gong, B.; Foroosh, H. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 9598–9607.
10. Lawin, F.J.; Danelljan, M.; Tosteberg, P.; Bhat, G.; Khan, F.S.; Felsberg, M. Deep Projective 3D Semantic Segmentation. In Proceedings of the Computer Analysis of Images and Patterns, Ystad, Sweden, 22–24 August 2017; pp. 55–107.
11. Boulch, A.; Saux, B.L.; Audebert, N. Unstructured point cloud semantic labeling using deep segmentation networks. In Proceedings of the Workshop on 3D Object Retrieval (3Dor '17). Eurographics Association, Goslar, Germany, 23 April 2017; pp. 17–24.
12. Tatarchenko, M.; Park, J.; Koltun, V.; Zhou, Q.-Y. Tangent Convolutions for Dense Prediction in 3D. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3887–3896.
13. Su, H.; Jampani, V.; Sun, D.; Maji, S.; Kalogerakis, E.; Yang, M.H.; Kautz, J. SPLATNet: Sparse Lattice Networks for Point Cloud Processing. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2530–2539.
14. Wu, B.; Wan, A.; Yue, X.; Keutzer, K. SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1887–1893.
15. Milioto, A.; Stachniss, C. RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 4213–4220.
16. Su, H.; Maji, S.; Kalogerakis, E.; Learned-Miller, E. Multi-view Convolutional Neural Networks for 3D Shape Recognition. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 945–953.
17. Feng, Y.; Zhang, Z.; Zhao, X.; Ji, R.; Gao, Y. GVCNN: Group-View Convolutional Neural Networks for 3D Shape Recognition. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 264–272.
18. Wang, C.; Pelillo, M.; Siddiqi, K. Dominant set clustering and pooling for multi-view 3D object recognition. In Proceedings of the British Machine Vision Conference 2017, London, UK, 4–7 September 2017; pp. 1–12.
19. Ma, C.; Guo, Y.; Yang, J.; An, W. Learning Multi-View Representation with LSTM for 3-D Shape Recognition and Retrieval. *IEEE Trans. Multimed.* **2019**, *21*, 1169–1182. [CrossRef]
20. Maturana, D.; Scherer, S. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; pp. 922–928.
21. Rethage, D.; Wald, J.; Sturm, J.; Navab, N.; Tombari, F. Fully-Convolutional Point Networks for Large-Scale Point Clouds. In Proceedings of the European Conference on Computer Vision ECCV 2018, Munich, Germany, 8–14 September 2018; pp. 596–611.
22. Graham, B.; Engelcke, M.; Maaten, L. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9224–9232.
23. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3D ShapeNets: A deep representation for volumetric shapes. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
24. Riegler, G.; Ulusoy, A.O.; Geiger, A. OctNet: Learning Deep 3D Representations at High Resolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6620–6629.

25. Wang, P.S.; Liu, Y.X.; Guo, Y.X.; Sun, C.Y.; Tong, X. O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis. *ACM Trans. Graph.* **2017**, *36*, 1–11. [CrossRef]
26. Xu, Y.; Hoegner, L.; Tuttas, S.; Stilla, U. Voxel- and Graph-based Point Cloud Segmentation of 3D Scenes Using Perceptual Grouping Laws. In Proceedings of the ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, Boston, MA, USA, 7–12 June 2017; pp. 43–50.
27. Li, Y.Y.; Pirk, S.; Su, H.; Qi, C.R.; Guibas, L.J. FPNN: Field Probing Neural Networks for 3D Data. In Proceedings of the NIPS'16: Proceedings of the 30th International Conference on Nrural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 207–315.
28. Le, T.; Duan, Y. PointGrid: A Deep Network for 3D Shape Understanding. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9204–9214.
29. Tchapmi, L.; Choy, C.; Armeni, I.; Gwak, J.; Savarese, S. SEGCloud: Semantic Segmentation of 3D Point Clouds. In Proceedings of the 2017 International Conference on 3D Vision (3DV), Qingdao, China, 10–12 October 2017; pp. 537–547.
30. Qi, C.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 77–85.
31. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 5105–5114.
32. Jiang, M.; Wu, Y.; Zhao, T.; Zhao, Z.; Lu, C. PointSIFT: A SIFT-like Network Module for 3D Point Cloud Semantic Segmentation. *arXiv* **2018**, arXiv:1807.00652.
33. Li, J.; Chen, B.M.; Lee, G.H. SO-Net: Self-Organizing Network for Point Cloud Analysis. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9397–9406.
34. Wang, Y.; Chao, W.; Garg, D.; Hariharan, B.; Campbell, M.; Weinberger, K. Pseudo-LiDAR From Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 8437–8445.
35. Yang, B.; Luo, W.; Urtasun, R. PIXOR: Real-time 3D Object Detection from Point Clouds. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7652–7660.
36. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. PointPillars: Fast Encoders for Object Detection From Point Clouds. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 8437–8445.
37. Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; Waslander, S.L. Joint 3D Proposal Generation and Object Detection from View Aggregation. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1–8.
38. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2009**, *20*, 61–80. [CrossRef] [PubMed]
39. Bruna, J.; Zaremba, W.; Szlam, A.; LeCun, Y. Spectral Networks and Locally Connected Networks on Graphs. *arXiv* **2013**, arXiv:1312.6203.
40. Simonovsky, M.; Komodakis, N. Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 29–38.
41. Landrieu, L.; Simonovsky, M. Large-Scale Point Cloud Semantic Segmentation with Superpoint Graphs. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4558–4567.
42. Landrieu, L.; Boussaha, M. Point Cloud Oversegmentation with Graph-Structured Deep Metric Learning. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 7432–7441.
43. Jiang, L.; Zhao, H.; Liu, S.; Shen, X.; Fu, C.-W.; Jia, J. Hierarchical Point-Edge Interaction Network for Point Cloud Semantic Segmentation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 10432–10440.
44. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2016**, arXiv:1609.02907.
45. Te, G.S.; Hu, W.; Zheng, A.M.; Guo, Z. RGCNN: Regularized Graph CNN for Point Cloud Segmentation. In Proceedings of the 26th ACM International Conference on Multimedia (MM '18), Seoul, Korea, 22–26 October 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 746–754.
46. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.; Bronstein, M.; Solomon, J. Dynamic Graph CNN for Learning on Point Clouds. *arXiv* **2018**, arXiv:1801.07829. [CrossRef]
47. Behley, J.; Garbade, M.; Milioto, A.; Quenzel, J.; Behnke, S.; Stachniss, C.; Gall, J. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 9296–9306.

48. Wu, B.; Zhou, X.; Zhao, S.; Yue, X.; Keutzer, K. SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 4376–4382.
49. Hu, Q.; Yang, B.; Xie, L.; Rosa, S.; Guo, Y.; Wang, Z.; Trigoni, N.; Markham, A. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 11105–11114.