information

Article

Local Transformer Network on 3D Point Cloud Semantic Segmentation

Zijun Wang [1,2], Yun Wang [1,2,*], Lifeng An [1], Jian Liu [1] and Haiyang Liu [1]

[1] Institute of Microelectronics, Chinese Academy of Sciences, Beijing 100029, China; wangzijun@ime.ac.cn (Z.W.); anlifeng@ime.ac.cn (L.A.); liujian@ime.ac.cn (J.L.); liuhaiyang@ime.ac.cn (H.L.)
[2] University of Chinese Academy of Sciences, Beijing 100049, China
* Correspondence: wangyun@ime.ac.cn

Abstract: Semantic segmentation is an important component in understanding the 3D point cloud scene. Whether we can effectively obtain local and global contextual information from points is of great significance in improving the performance of 3D point cloud semantic segmentation. In this paper, we propose a self-attention feature extraction module: the local transformer structure. By stacking the encoder layer composed of this structure, we can extract local features while preserving global connectivity. The structure can automatically learn each point feature from its neighborhoods and is invariant to different point orders. We designed two unique key matrices, each of which focuses on the feature similarities and geometric structure relationships between the points to generate attention weight matrices. Additionally, the cross-skip selection of neighbors is used to obtain larger receptive fields for each point without increasing the number of calculations required, and can therefore better deal with the junction between multiple objects. When the new network was verified on the S3DIS, the mean intersection over union was 69.1%, and the segmentation accuracies on the complex outdoor scene datasets Semantic3D and SemanticKITTI were 94.3% and 87.8%, respectively, which demonstrate the effectiveness of the proposed methods.

Keywords: transformer; point clouds; semantic segmentation; self-attention; feature extraction

Citation: Wang, Z.; Wang, Y.; An, L.; Liu, J.; Liu, H. Local Transformer Network on 3D Point Cloud Semantic Segmentation. *Information* **2022**, *13*, 198. https://doi.org/10.3390/info13040198

Academic Editor: Gholamreza Anbarjafari (Shahab)

Received: 19 February 2022
Accepted: 2 April 2022
Published: 14 April 2022

1. Introduction

The semantic segmentation of 3D point clouds is one of the key problems in the perception of environments in the research on robotics [1] and automatic driving [2]. The use of local and global geometric attributes of a point to generate an effective feature description of the point, thereby improving the accuracy of semantic segmentation, has always been the focus and challenge in this field.

Convolutional neural networks (CNNs) have achieved great success in 2D imaging, with researchers considering how to use a mature CNN network to analyze 3D point clouds. However, unlike 2D images, point clouds are unordered and irregular, which make handling 3D point clouds using a convolutional neural network directly impossible. Some methods [3–7] project the 3D point clouds onto the 2D plane, generating a bird's eye view (BEV) image, a range view (RV) image, and other intermediate regular representations, and then use them in a convolutional neural network. This kind of method depends on the choice of projection angle, which cannot make full use of accurate spatial and structural information and causes a loss of geometric information in the projection process. The method based on discretization converts the point clouds into a discretized representation, such as a voxel or lattice, and then processes it using a three-dimensional convolution network. However, it is sensitive to voxel size. When the voxel is large, it causes information loss and affects the segmentation accuracy. When the voxel is small, it leads to a sharp increase in the number of calculations required and affects the real-time performance. The point-based method extracts features directly from irregular point clouds without

any preprocessing. It has gradually become the mainstream method used in point cloud semantic segmentation. However, the point-based method also has some problems, such as the poor scalability of the point sampling method to the scale of the point cloud, and the inability to effectively learn local features.

With the successful application of a self-attention mechanism in the field of natural language processing (NLP), some studies have considered the applications of the transformer structure to the fields of image and point cloud processing, etc. The input of a transformer is usually a sequence, and position embedding information needs to be added. For point cloud data, each point has a unique coordinate value, which can be directly used as the position embedding information. Zhao et al. [8] proposed the point transformer and proved that a network structure based entirely on self-attention can effectively solve a point cloud task.

The transformer method has been broadly used for object-wise tasks, such as classification and partial segmentation. Inspired by these works, in this study, we use a transformer network for point-wise large-scale point cloud segmentation. We propose a novel multi-scale transformer network for both local and global feature extraction.

The network is based on an encoder–decoder structure. Each encoder layer consists of two parallel local transformers. In the front of the encoder layers, local features can be obtained because of the locality of the local transformer. After random down-sampling of the points, the receptive field of each point becomes larger and every point contains high-level features. Therefore, at the later encoder layers of the encoder, the global features can be easily obtained by the local transformer structure, making full use of the transformer having no inductive bias property.

In contrast to a previous transformer structure with one attention weight matrix, which can find the similarities between point features, we propose two different key matrices to obtain two attention weight matrices in the local transformer structure, which can not only focus on the feature similarity between points but also focus on the local geometric structure relationship. The results of the visualization show that better segmentation results can be obtained between objects with very similar geometries. We also propose two fusion strategies to make full use of the distribution diversity of these two attention weight matrices.

In order to improve the segmentation performance at the semantic edges of multiple objects, we propose a novel neighbor selection method called cross-skip selection, which is very suitable for the parallel encoder layer, and can expand the receptive field of each point without increasing the number of calculations required, and capture more abundant geometric and semantic information.

We then verified our method on open datasets S3DIS and Semantics3D. The best mean class-wise intersection over union (mIoU) on the S3DIS dataset was 69.1% and on the Semantic3D dataset was 75.7%, which are better than those of most benchmark methods.

Our contributions can be summarized as follows:

1. We propose a novel multi-scale transformer network to learn local context information and global features, which makes applying the transformer on more sophisticated tasks from the large-scale point cloud datasets possible.
2. In order to obtain the feature similarity and local geometry relationship between points, we propose two different key matrices to obtain two attention weight matrices in the local transformer structure, and propose two different fusion strategies to fuse them.
3. We also propose a novel neighbor selection method called cross-skip selection to obtain more accurate results on the junction of multiple objects.

The rest of the paper is organized as follows. Section 2 presents related work on 3D point cloud semantic segmentation. Section 3 presents our proposed approach, including the network architecture (Section 3.1), neighbor embedding module (Section 3.2), feature extraction module based on a transformer (Section 3.3), local transformer structure (Section 3.4), parallel encoder layer with cross-skip selection (Section 3.5), and decoder layer (Section 3.6).

Section 4 presents our experiments and analysis. Section 5 concludes our work and presents some future work.

2. Related Work

According to the different forms of input point clouds, the semantic segmentation methods of a point cloud can be divided into the projection-based method, the discretization-based method, the point-based method, etc.

The projection-based method is used to project the 3D point cloud onto the 2D plane. Rangenet + + [9] first converted the point cloud into a range image, then used the full convolution network to deal with that, and then mapped the result of the semantic segmentation to the original point cloud. Squeezeseg [10] converted the point cloud to the front view through spherical projection, using the SqueezeNet network for semantic segmentation, and then used a conditional random field (CRF) to refine the results. Liong et al. [11] proposed a multi-view fusion network (AMVNet), which projected the point cloud onto the range view (RV) image and bird's eye view (BEV) image, and combined the advantages of using two different views of RV and BEV.

VoxNet [12] is a typical discretization-based method. The network divided the point cloud into regular voxels and then extracted the features of these voxels through a 3D convolution operation. However, the point cloud is sparse, and the proportion of non-empty voxels is very small. It is very inefficient to use a dense convolution neural network on spatial sparse data. Graham et al. [13] improved this by proposing a new sparse convolution operation that can process spatial sparse data more effectively.

PointNet [14] was the pioneer work that directly consumed point clouds, and could obtain per-point features by concatenating features learned by the shared multilayer perceptron (MLP) and global features learned by max pooling function. However, PointNet cannot effectively obtain local features and ignores the local context information. The PointNet + + [15] network made some improvements. It paid attention to the relationship between the central point and its neighbors but ignored the relationship between each neighbor pair. Wu et al. [16] proposed a new convolution operation, which defined the convolution kernel as a nonlinear function composed of a weight function and a density function. Zhao et al. [17] proposed a network called PointWeb, which can specify the feature of each point based on the local region characteristics for better representing the region. KPConv [18] defined an explicit convolution kernel, which is composed of fixed or flexible core points. The different weights of influence for each neighbor point on the core point adaptively depends on the distance between them. Hu et al. [19] used MLP to learn the attention weight score of each point and then obtained the weighted local feature map. Finally, the maximum pooling function was used to calculate the central point feature on the weighted local feature map.

In addition, there are some other methods. For example, the dynamic graph convolutional neural network (DGCNN) [20] uses graph networks to capture the local geometric features of a point cloud and dynamically updates a graph to learn the different scale features. Wang et al. [21] proposed a graph attention convolution (GAC), in which the kernels can be dynamically carved into specific shapes to adapt to the structure of an object and generate more accurate representations of local features of points.

Recently, the transformer network has already achieved tremendous success in the language domain; thus, most researchers want to investigate whether it can be applied to computer vision tasks. It is very encouraging that some methods [22] showed that a pure transformer applied directly to sequences of image patches can perform very well on image classification tasks. After that work was conducted, research on transformers in computer vision became popular. The operator of the transformer network is invariant to permutation, making it particularly appropriate for point clouds, since a point cloud is permutation-invariant.

Some recent work considered how to effectively extract local and global features at the same time. LocalViT [23] aimed to combine the local performance of a convolutional

neural network and the global connectivity of a transformer structure. However, using a transformer structure to learn the long-distance dependence between points requires very high memory and expensive computing costs, which make it difficult to deal with large-scale 3D point cloud data. In this paper, we propose a local transformer structure and a novel neighborhood sampling method for local feature extraction; it can not only effectively focus on local features, but also reduce the computational costs to adapt to the semantic segmentation tasks of large-scale outdoor datasets.

3. Proposed Approach

3.1. Network Architecture

The overall network structure adopts the typical encoder–decoder structure [24]. The encoder is composed of parallel encoder layers (Section 3.5), which consist of two local transformer structures (Section 3.4). The local transformer structure is composed of a neighbor embedding module (Section 3.2) and a feature extraction module based on a transformer (Section 3.3). As shown in Figure 1, after each parallel encoder layer, random down-sampling is used to reduce the number of points and then to stack parallel encoder layers. As the encoder layers are stacked, the semantic features of each point become more and more abstract and contain more contextual information. We set up a different number of encoder layers for different datasets. Then, the number of points is recovered through the decoder layers. In this paper, we use a distance-based weighted linear interpolation up-sampling operation (Section 3.5) to propagate the features from a sparse point cloud to a dense point cloud to predict point-wise labels. The whole network structure adopts a residual connection.

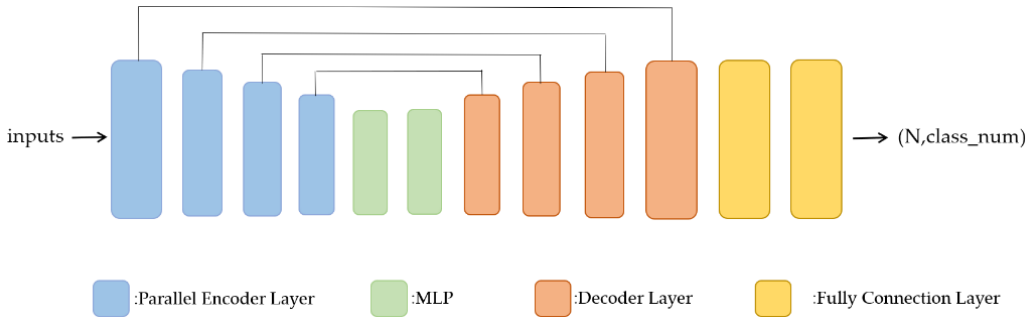


Figure 1. The proposed network architecture.

3.2. Neighbor Embedding Module

Point-wise local geometric properties, such as the normal vector (the normal vector of each point is the normal direction of the plane fitted by it and its neighbors), are very useful for discovering the similarity between points, since the normal vector of the surface of an object changes continuously. Before the data are input into the network, we compute the normal vector of S3DIS and combine it with their corresponding point coordinates as a rich feature representation. We use principal components analysis (PCA) to compute the normal of every point (as shown in Figure 2). For each point, we choose its M neighbors $P : \{p_1, p_2, \ldots p_M\} \in R^{M\times 3}$. We want to fit a plane using the M neighbors, and then, we can calculate the normal vector of the point. The normal vector is solved by minimizing an objective function as follows:

$$\min_{c,n,||n||=1} \sum_{i=1}^{M} ((p_i - c)^T n)^2, \tag{1}$$

$$c = \frac{1}{M} \sum_{i=1}^{M} p_i, \tag{2}$$

where c is a coordinate of the center of all neighbor points and n is the normal vector to be solved.

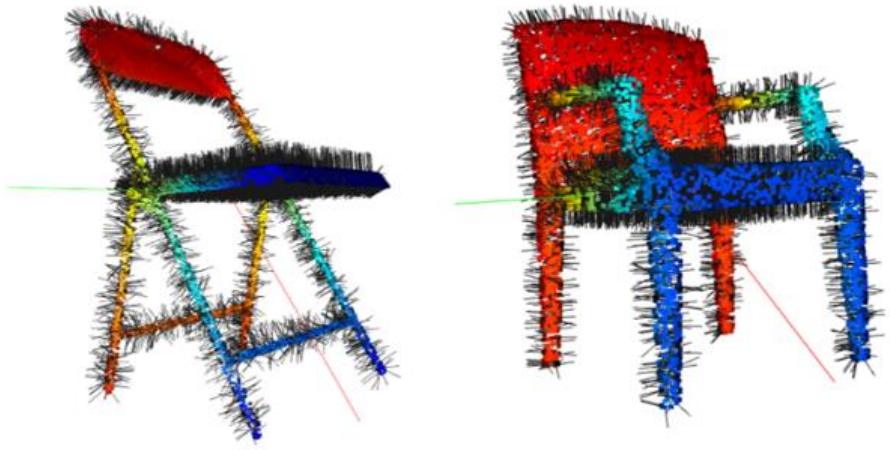


Figure 2. The normal vector of every point (results of the visualization on two different chairs).

The input point cloud is represented as a coordinate matrix $P : \{p_1, p_2, \ldots p_N\} \in R^{N\times 3}$ and its feature matrix $F : \{f_1, f_2, \ldots f_N\} \in R^{N\times D}$. N and D are the number of points and the feature dimensions, respectively. The original input features of the points are concatenated by the normalized x-y-z coordinates, raw RGB, and surface normal information.

First, we encode the local region in the neighbor embedding module, as shown in Figure 3, similar to a reference point x_i, and find its $2K$ neighbors, encoding each neighbor point to obtain its position embedding $G : \{g_{i1}, g_{i2}, \ldots g_{i*2k}\} \in R^{2K\times D'}$, as shown in Formula (3). g_{ik} is the position embedding of the k-th neighbor.

$$g_{ik} = (p_i - p_{ik}) \oplus ||p_i - p_{ik}||. \tag{3}$$

where $\oplus$ stands for the concatenation of $(p_i - p_{ik})$ and $||p_i - p_{ik}||$ on the feature dimension, p_i is the coordinate of the point x_i, p_{ik} is the coordinate of the k-th neighbor, and $||\cdot||$ is the l_1 norm.

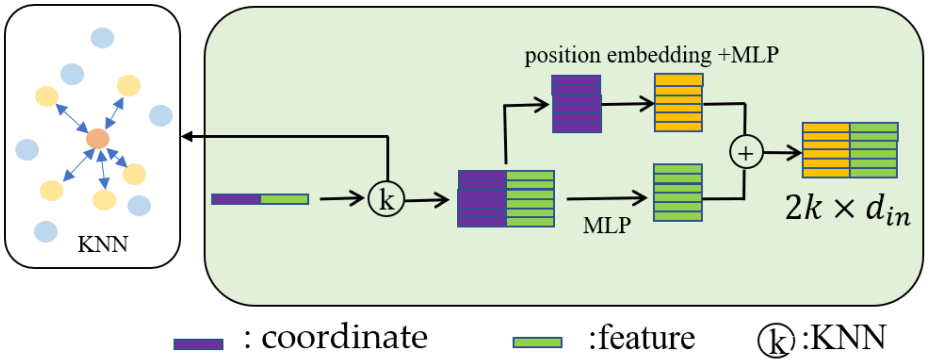


Figure 3. Neighbor embedding module.

Then, the local region feature matrix $I_{in} : \{I_{i1}, I_{i2}, \ldots I_{i*2k}\}$ can be obtained by concatenating the positional embedding g_{ik} and per-point original features f_{ik} as follows:

$$I_{ik} = MLP(g_{ik}) \oplus MLP(f_{ik}). \tag{4}$$

where $MLP(\cdot)$ is a multilayer perceptron applied to the positional embedding g_{ik} and features f_{ik}.

3.3. Feature Extraction Module based on a Transformer

3.3.1. Naïve Transformer Structure

Using the neighbor embedding module, we can obtain the local region feature matrix: $I_{in} \in R^{2k\times d_{in}}$ (d_{in} is the dimension of the features). It is then input into a transformer

structure to obtain the weighted features of the reference point. In the naïve transformer structure, Q, K, V are obtained based on I_{in}, as shown in Formula (5).

$$Q = BN(MLP_1(I_{in})),\ K = BN(MLP_2(I_{in})),\ V = BN(MLP_3(I_{in})),\ Q, K, V \in R^{k \times d} \quad (5)$$

where $BN(MLP(\cdot))$ stands for a batch normalization operation after a multilayer perceptron. The batch normalization normalizes the batch input on the feature dimension. It can make the distribution of input data in each layer of the network relatively stable and accelerates the learning speed of the model. d is the dimension of Q, K, V. We set the dimensions of Q, K, V to be equal.

The attention score is computed as the inner product between Q and K. The attention weight matrix can be written as follows:

$$W = (\widetilde{w}_{ij}) = Q \cdot K^T. \quad (6)$$

Additionally, a scale and a normalization operation, which is able to obtain measurements of the similarity between any two points in this local region, can be used:

$$\overline{w}_{ij} = \frac{\widetilde{w}_{ij}}{\sqrt{d}}, \quad (7)$$

$$w_{ij} = softmax(\overline{w}_{ij}) = \frac{\exp(\overline{w}_{ij})}{\sum_k \exp(\overline{w}_{i,k})}. \quad (8)$$

Next, the normalized attention weight matrix is applied on the value matrix V to obtain a weighted local feature matrix, which can automatically assign more attention to the useful features. The weighted local feature matrix can be written as follows:

$$I_w = W \cdot V,\ I_w \epsilon R^{2k \times d}. \quad (9)$$

Then, we use batch normalization and non-linear activation on this weighted local feature matrix. After that, we concatenate the activated features matrix and the original input features matrix as follows:

$$I_1 = LBR(I_w) \oplus I_{in},\ I_1 \in R^{2k \times 2d_{in}}. \quad (10)$$

where LBR is the normalization and non-linear activation operation and $\oplus$ is the concatenation operation.

We use a symmetric operation such as max pooling to generate the reference point features, which can represent this local region. It is formally defined as follows:

$$I_{out} = MLP(maxpooling(I_1)),\ I_{out} \in R^{1 \times d_{out}}. \quad (11)$$

Figure 4 provides an illustration of the naïve transformer structure. Through this transformer, the updated features of each reference point adaptively aggregate the features of its neighbor points.

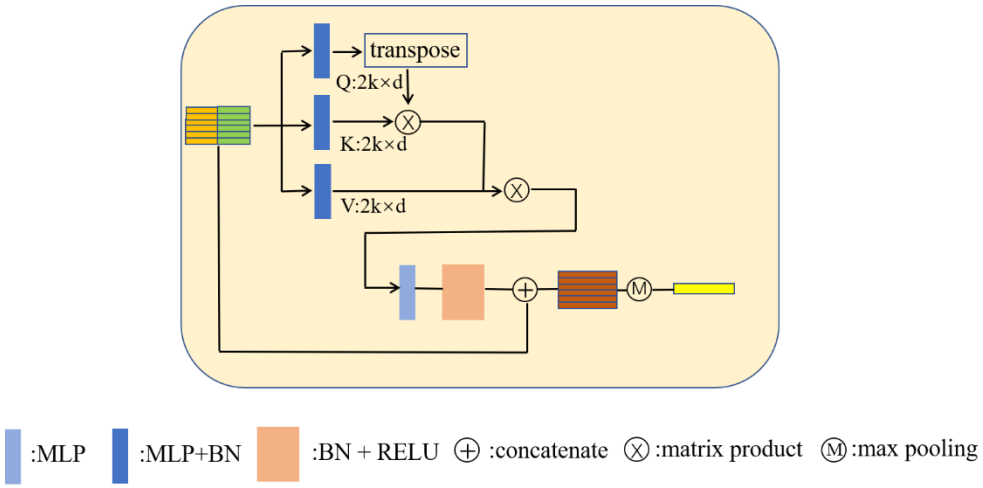


Figure 4. Naïve transformer structure.

3.3.2. Improve Transformer Structure

We make some improvements to the naïve local transformer structure. Consider that, during the beginning of the network, where each point has not learned enough features yet to obtain the similarity between points, the geometric relationships between points become more significant than the features. Therefore, we propose two different key matrices for obtaining two attention weight matrices. The first key matrix is the same as the naïve local transformer structure, which is obtained by the input feature matrix I_{in}, as shown in Formula (12):

$$K_1 = BN(MLP(I_{in})), K_1 \in R^{2K \times d}. \tag{12}$$

The attention weight matrix can be written as Formula (13), which can obtain the feature similarity between points:

$$W_1 = (w_1)_{ij} = Q \cdot K_1^T. \tag{13}$$

The query matrix Q can be obtained by Formula (5).

The second key matrix, which is obtained by the neighbor position embedding matrix $G : \{g_{i1}, g_{i2}, \ldots g_{i*4k}\} \in R^{2K \times D'}$, takes the local geometry relationship into account. The second key matrix can be written as follows:

$$K_2 = BN(MLP(G_{in})), K_2 \in R^{2K \times d}. \tag{14}$$

The second attention weight matrix is as follows and can pay more attention to the local geometry relationship between points than W_1:

$$W_2 = (w_2)_{ij} = Q \cdot K_2^T. \tag{15}$$

When the two attention weight matrices are obtained, the most intuitive method for combining them is to add them, which can be shown as follows:

$$W_{add} = W_1 + W_2. \tag{16}$$

The next operations are the same as the naïve local transformer structure. Finally, we can obtain the weighted local feature matrix.

$$I_w = W_{add} \cdot V, I_w \in R^{2K \times d}. \tag{17}$$

where V is the key matrix, which can be obtained using Formula (5).

However, if we simply add the two attention weight matrices together, we cannot make full use of the distribution diversity of the two attention weight matrices. We propose

another fusion method at the feature level. Specifically, we multiply the two scaled and normalized attention weight matrices with the value matrix, as shown in (18) and (19), respectively, and then concatenate the weighted local feature matrices, as shown in (20).

$$I_{w1} = W_1 \cdot V, I_{w1} \in R^{2K \times d}, \tag{18}$$

$$I_{w2} = W_2 \cdot V, I_{w2} \in R^{2K \times d}, \tag{19}$$

$$I_w = I_{w1} \oplus I_{w2}, I_w \in R^{2K \times 2d}. \tag{20}$$

where V is the key matrix, which can be obtained using Formula (5).

By fusing at the feature level, we can obtain weighted feature matrices I_{w1} and I_{w2} with two different distributions, which can inherently capture the features and geometric relationships between points.

The updated point feature I_{out} can finally be obtained using the operations in (10) and (11) on I_w. The improved transformer structures are shown in Figure 5:

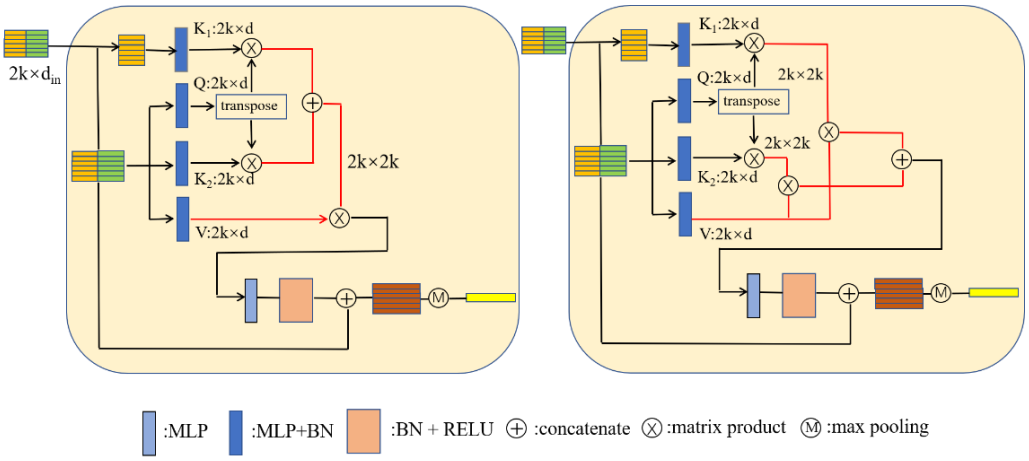


Figure 5. Improved transformer structure. (**Right**): the two attention weight matrices added. (**Left**): the two attention weight matrices fused at the feature level. The red line indicates the difference between them.

We also visualized the attention weight matrices, presented in Appendix A. From the results of the visualization, we can note that, at the first few layers, W_{add} shows a locality (the value of the diagonal is significantly different from the others) and, at the last two layers, it shows a discretized distribution, which can be used to find points with key semantic information. When we fuse the attention weight matrices at the feature level, at every layer, W_1 and W_2 focus on the feature similarities and the geometry relationship between points, respectively. Therefore, they have different distributions.

3.4. Local Transformer Structure

The local transformer structure is composed of a neighbor embedding module and a transformer structure, which is shown in Figure 6.

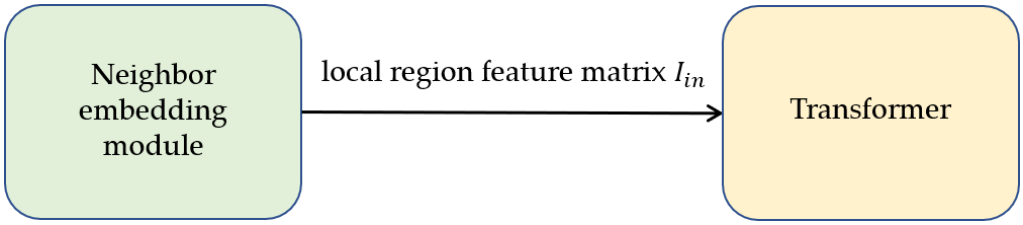


Figure 6. Local transformer structure.

3.5. Parallel Encoder Layer with Cross-Skip Selection

The distance and feature similarities between each point and its neighbors are not necessarily positively correlated, especially at the junction of multiple objects. Therefore, points with similar semantic structures may have greater Euclidean distances and vice versa. As shown in Figure 7, the distance between c_1 and c_1' is greater than that between c_1 and c_2, while c_1 and c_1' belong to class 1 and c_2 belongs to class 2.

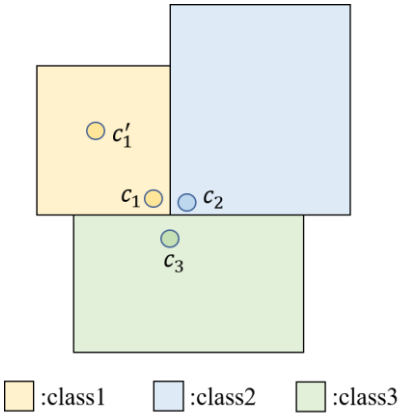


Figure 7. Junction of multiple objects.

We propose a cross-skip selection method to obtain neighbors in the neighbor embedding module. The method is as follows: find the nearest $4K$ neighbor points of each point and divide them into four groups k_1, k_2, k_3, k_4 with the same number. Each group contains K points. Concatenate the point feature in k_1 and in k_3, k_2, and k_4, and then encode the two original point feature matrices in the neighbor embedding module to obtain the local region feature matrices I_{in1} and I_{in2}.

In order to adapt to the cross-skip selection method, we propose a parallel encoder layer composed of two parallel local transformer structures, as shown in Figure 8. The two input matrices of the same dimension I_{in1} and I_{in2} are composed of different neighbor points selected based on cross-skip selection. This structure allows each point to obtain a larger receptive field without increasing the dimension of the input matrices I_{in1} and I_{in2}. Concatenate the two update feature vectors I_{out1} and I_{out2} to obtain the final output I_{out}.

$$I_{out} = I_{out1} \oplus I_{out2}. \tag{21}$$

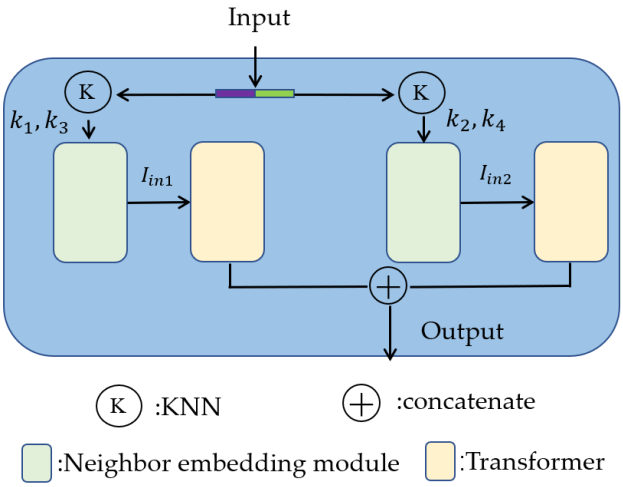


Figure 8. Parallel encoder layer.

3.6. Decoder Layer

To finally obtain each original point feature, we simply adopt a distance-based weighted linear interpolation up-sampling operation, shown in Figure 9. Let P_{l-1} and P_l be the coordinate set of the $(l-1)$-th layer and (l)-th layer at the decoder block, respectively. To obtain the feature of point p_i at P_l, we search the nearest three neighbors ($t = 3$) of p_i in P_{l-1}. The coordinates of neighbors are $p_{it} = \{p_{i1}, p_{i2}, p_{i3}\}$, and we obtain the influence of each neighbor point on p_i based on the distance between the point p_i and $p_{it} = \{p_{i1}, p_{i2}, p_{i3}\}$, as follows:

$$M_{ij} = (m_{ij}) = MLP(||p_i - p_{ij}||). \tag{22}$$

where M_{ij} is the influence weight and p_{ij} is the coordinate of the j-th neighbor of the point p_i.

Finally, the feature of p_i is as follows:

$$f_i = \sum_{j=1}^{3} m_{ij} p_{ij}. \tag{23}$$

where m_{ij} is the weight of influence for each neighbor point obtained by Formula (22).

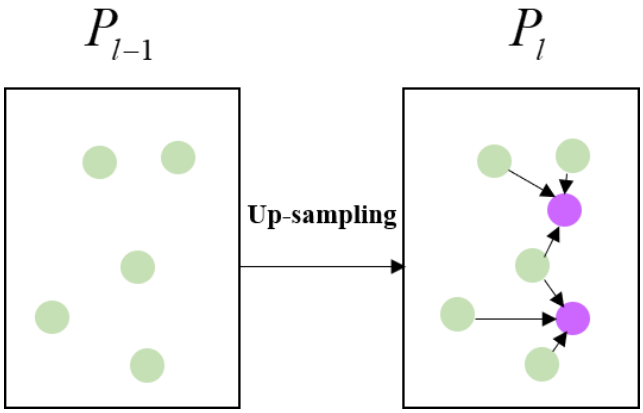


Figure 9. Decoder layer: distance-based weighted linear interpolation up-sampling.

4. Experiments and Analysis

4.1. Datasets

S3DIS [25]: the Stanford 3D Large-Scale Indoor Spaces (S3DIS) dataset addresses semantic tasks with pixel-level semantic annotations developed by Stanford University. It consists of six areas, each of which contains 13 categories such as ceiling, floor, and wall. We evaluated our network using two methods: (1) six-fold cross-validation and (2) area 5 testing. Moreover, we used the mean class-wise intersection over union (mIoU), the mean of class-wise accuracy (mAcc), and the overall point-wise accuracy (OA) as evaluation metrics.

Semantic3D [26]: the Semantic3D dataset is composed of 30 non-overlapping outdoor point cloud scenes, of which 15 scenes are used for training and other scenes are used for online testing. The dataset contains eight categories. The scenes cover rural, urban, and suburban areas, and each scene covers sizes of $160 \times 240 \times 30$. In addition to 3D coordinates, the dataset provides RGB values and intensity values. We used the mean class-wise intersection over union (mIoU) and the overall point-wise accuracy (OA) as evaluation metrics.

SemanticKITTI [27]: the SemanticKITTI dataset is composed of 21 sequences and 43,552 densely annotated laser scanning frames. Among these, sequences 00–07 and 09–10 are used for training, sequence 08 is used for verification, and sequences 11–21 are used for online testing. The raw data contain the 3D coordinate information of the points. We used the mean class-wise intersection over union (mIoU) as an evaluation metric.

4.2. Implementation Details

When verifying the proposed model based on the S3DIS dataset, we first obtained the normal vector of each point as the original feature. We set the number of encoder layers as 7. After each layer extracts the characteristics of each point through the encoder layer, random down-sampling is used to reduce the number of points. The random down-sampling is more efficient than other down-sampling methods, which have a high calculation cost and high GPU memory requirements. We set the sampling ratio to [2, 2, 4, 4, 4, 4, 4] and the output dimension of each layer to [16, 64, 256, 256, 512, 512]. Through distance-based weighted interpolation linear up-sampling, three nearest neighbors are selected to restore the feature of the input points. Finally, three fully connected layers are stacked to obtain the output with a category number dimension. When verifying the model on the SemanticKITTI and Semantic3D datasets, because the number of points in each frame is very large, calculating the normal vector of each point consumes a very large amount of memory, so we only calculate the normal features in an indoor dataset.

In addition, we used a residual connection to retain more point feature information. In this paper, the Adam optimizer and weighted cross entropy loss based on inverse frequency were used for training. All experiments used Tensorflow as the platform and applied an NVIDIA Corporation gp102 (Titan XP) GPU.

4.3. Evaluation Metrics

For the S3DIS dataset, the mean class-wise intersection over union (*mIoU*), mean class Accuracy (*mAcc*), and Overall Accuracy (*OA*) of the total 13 classes were compared. For Semantic 3D, the mIoU and OA were used as the evaluation metrics. For SemanticKITTI, we used mIoU. The evaluation metrics can be defined as follows:

$$mIoU = \frac{1}{k+1}\sum_{i=0}^{k}\frac{p_{ii}}{\sum_{j=0}^{k}p_{ij}+\sum_{j=0}^{k}p_{ji}-p_{ii}}, \tag{24}$$

$$OA = \frac{1}{N}\sum_{i=0}^{k}p_{ii}, \tag{25}$$

$$mAcc = \frac{1}{k+1}\sum_{i=0,j\neq i}^{k}\frac{p_{ii}}{p_{ij}} \tag{26}$$

where $k+1$ is the number of classes, i is the ground-truth label, j is the prediction label, p_{ij} is the number of the samples that belong to i but are mistakenly predicted as j, p_{ji} is the number of the samples that belong to j but are mistakenly predicted as i, p_{ii} is the number of correctly predicted samples, and N is the total number of samples.

4.4. Experimental Results

We used area 5 of S3DIS for testing and the other areas for training. The results are shown in Table 1. Table 2 shows the results with six-fold cross-validation. Tables 3 and 4 show the results on the Semantic3D and SemanticKITTI datasets, respectively. The results show that the proposed method is better than most benchmark models. The results of the visualization on S3DIS and SemanticKITTI are shown in Figures 10 and 11.

Table 1. Segmentation results on area 5 of S3DIS (add: the two attention weight matrices added; con: the two attention weight matrices fused at the feature level).

Methods	OA	mAcc	mIoU	Ceiling	Floor	Wall	Beam	Column	Window	Door	Table	Chair	Sofa	Bookcase	Board	Clutter
TangentConv [28]	-	62.2	52.6	90.5	97.7	74.0	0.0	20.7	39.0	31.3	77.5	69.4	57.3	38.5	48.8	39.8
PointCNN [29]	85.9	63.9	57.3	92.3	98.2	79.4	0.0	17.6	22.8	62.1	74.4	80.6	31.7	66.7	62.1	56.7
SPG [30]	86.4	66.5	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
PointWeb [17]	87.0	66.6	60.3	92.0	**98.5**	79.4	0.0	21,1	59.7	34.8	76.3	88.3	46.9	69.3	64.9	52.5
KPConv [18]	-	72.8	**67.1**	92.8	97.3	**82.4**	0.0	23.9	58.0	**69.0**	81.5	**91.0**	**75.4**	**75.3**	66.7	**58.9**
BAAF-Net [31]	**88.9**	**73.1**	65.4	**92.9**	97.9	82.3	0.0	23.1	**65.5**	64.9	78.5	87.5	61.4	70.7	**68.7**	57.2
Ours(add)	87.6	71.9	64.1	92.8	97.4	79.9	0.0	22.6	59.4	52.7	77.0	87.6	73.3	70.4	66.8	53.1
Ours(con)	87.8	72.1	63.7	91.8	97.7	82.1	0.0	26.9	58.6	51.7	78.8	86.6	62.0	70.8	68.5	52.4

Table 2. Quantitative results on the S3DIS dataset (six-fold cross validation) (add: the two attention weight matrices added; con: the two attention weight matrices fused at the feature level).

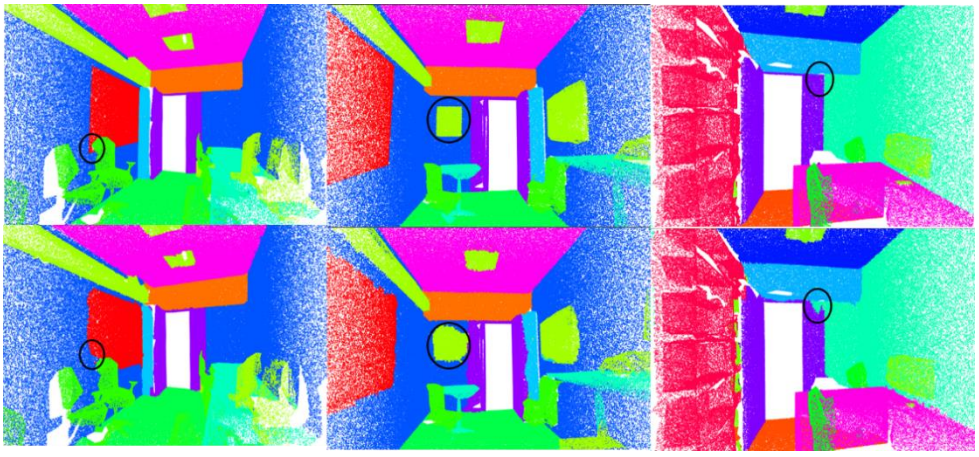
Methods	OA	mAcc	mIoU	Ceiling	Floor	Wall	Beam	Column	Window	Door	Table	Chair	Sofa	Bookcase	Board	Clutter
SPG [30]	85.5	73.0	62.1	89.9	95.1	76.4	62.8	47.1	55.3	68.4	**73.5**	69.2	63.2	45.9	8.7	52.9
PointWeb [17]	87.3	76.2	66.7	93.5	94.2	80.8	52.4	41.3	64.9	68.1	71.4	67.1	50.3	62.7	62.2	58.5
KPConv [18]	-	79.1	70.6	93.6	92.4	**83.1**	63.9	**54.3**	**66.1**	**76.6**	64.0	57.8	**74.9**	**69.3**	61.3	60.3
FKAConv [32]	-	-	68.4	**94.5**	**98.0**	82.9	41.0	46.0	57.8	74.1	71.7	77.7	60.3	65.0	55.0	**65.5**
SCF-Net [33]	88.4	82.7	71.6	93.3	96.4	80.9	**64.9**	47.4	64.5	70.1	71.4	81.6	67.2	64.4	**67.5**	60.9
BAAF-Net [31]	**88.9**	**83.1**	**72.2**	93.3	96.8	81.6	61.9	49.5	65.4	73.3	72.0	**83.7**	67.5	64.3	67.0	62.4
Ours(add)	87.4	80.1	68.8	92.8	97.0	80.0	58.2	48.5	62.4	68.7	71.7	70.2	58.9	63.3	65.6	57.3
Ours(con)	87.7	80.1	69.1	93.2	96.8	80.4	56.5	48.0	63.4	69.8	71.5	69.4	63.0	64.0	64.0	58.7

Table 3. Quantitative results on the Semantic3D dataset (add: the two attention weight matrices added; con: the two attention weight matrices fused at the feature level).

Methods	mIoU	OA	Man-Made	Natural.	High Veg.	Low veg.	Buildings	Hard Scape	Scanning Art.	Cars
ShellNet [34]	69.3	93.2	96.3	90.4	83.9	41.0	94.2	34.7	43.9	70.2
KPConv [18]	74.6	92.9	90.9	82.2	84.2	47.9	94.9	40.0	**77.3**	79.7
RGNet [35]	74.7	94.5	97.5	93.0	88.1	48.1	94.6	36.2	72.0	68.0
RandLA-Net [19]	77.4	94.8	95.6	91.4	86.6	51.5	**95.7**	**51.5**	69.8	79.7
BAAF-Net [31]	75.4	**94.9**	**97.9**	**95.0**	70.6	**63.1**	94.2	41.6	50.2	**90.3**
RFCR [36]	**77.8**	94.3	94.2	89.1	85.7	54.4	95.0	43.8	76.2	83.7
Ours(add)	74.4	94.0	96.7	92.4	85.6	50.5	93.5	31.4	63.8	81.2
Ours(con)	75.7	94.3	97.0	93.4	**88.2**	49.9	94.1	34.8	67.8	80.6

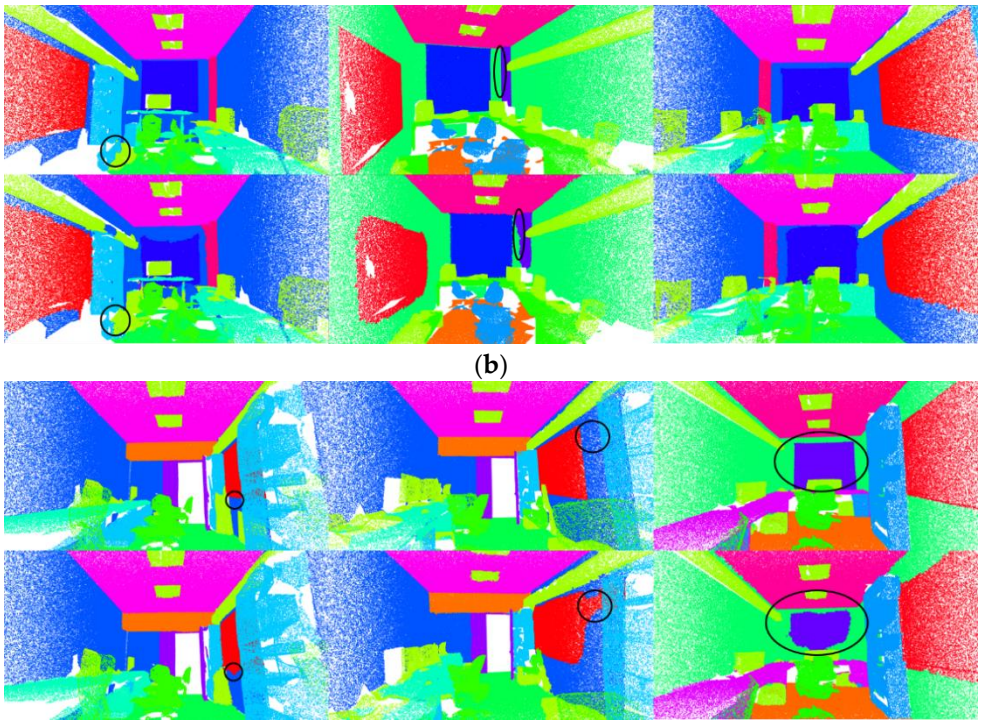
Table 4. Quantitative results on the SemanticKITTI dataset (add: the two attention weight matrices added; con: the two attention weight matrices fused at the feature level).

Methods	mIoU	Road	Sidewalk	Parking	Other-Ground	Building	Car	Truck	Bicycle	Motorcycle	Other-vehicle	Vegetation	Trunk	Terrain	Person	Bicyclist	Motorcyclist	Fence	Pole	Traffic-sign
SqueezeSge [10]	29.5	85.4	54.3	26.9	4.5	57.4	68.8	3.3	16.0	4.1	3.6	60.0	24.3	53.7	12.9	13.1	0.9	29.0	17.5	24.5
TangentConv [28]	40.9	83.9	63.9	33.4	15.4	83.4	90.8	15.2	2.7	16.5	12.1	79.5	49.3	58.1	23.0	28.4	**8.1**	49.0	35.8	28.5
SqueezeSegV2 [37]	39.7	88.6	67.6	45.8	17.7	73.7	81.8	13.4	18.5	17.9	14.0	71.8	35.8	60.2	20.1	25.1	3.9	41.1	20.2	36.3
DarkNet53Seg [27]	49.9	**91.8**	**74.6**	**64.8**	27.9	84.1	86.4	25.5	24.5	24.5	22.6	78.3	50.1	64.0	36.2	33.6	4.7	55.0	38.9	52.2
RandLA-Net [19]	53.9	90.7	73.7	60.3	20.4	86.9	**94.2**	40.1	26.0	25.8	**38.9**	81.4	61.3	66.8	**49.2**	**48.2**	7.2	56.3	49.2	47.7
PolarNet [4]	**54.3**	90.8	74.4	61.7	21.7	**90.0**	93.8	22.9	40.3	**30.1**	28.5	**84.0**	**65.5**	**67.8**	43.2	40.2	5.6	**67.8**	**51.8**	**57.5**
Ours(add)	49.8	89.7	71.2	58.1	**29.2**	86.6	92.4	**40.6**	**44.1**	21.8	29.2	79.6	60.3	62.1	45.5	44.1	3.5	54.9	46.0	36.9
Ours(con)	49.3	89.4	69.7	57.4	5.7	85.7	92.6	28.7	19.3	27.2	26.5	80.0	**61.3**	60.6	47.7	45.2	1.0	51.2	48.4	39.7



(**a**)

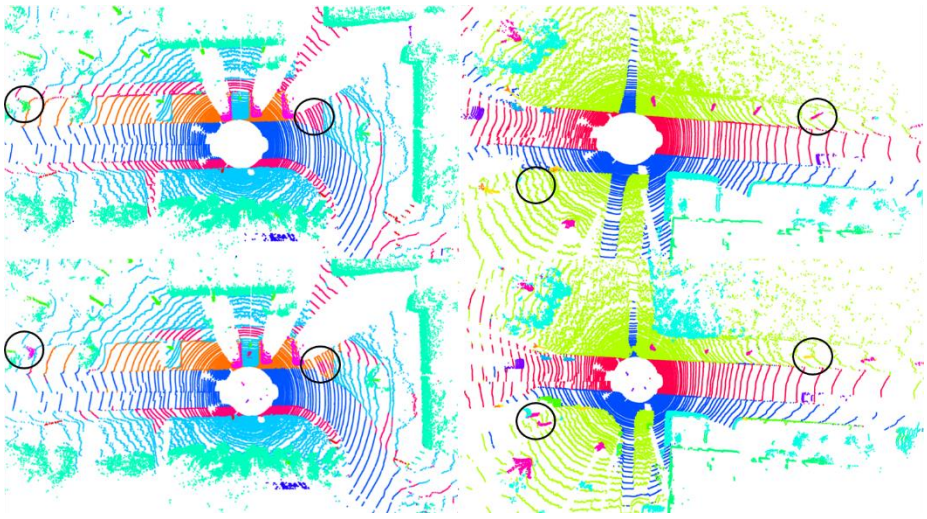
Figure 10. *Cont.*



(**b**)

(**c**)

Figure 10. Results of the visualization on S3DIS. First row is the ground truth. Second row shows the prediction results, where an incorrect prediction is circled. (**a**–**c**) Three different scenarios in the S3DIS dataset. The figure shows the prediction results of nine different scenarios in total.



(**a**)

Figure 11. *Cont*.

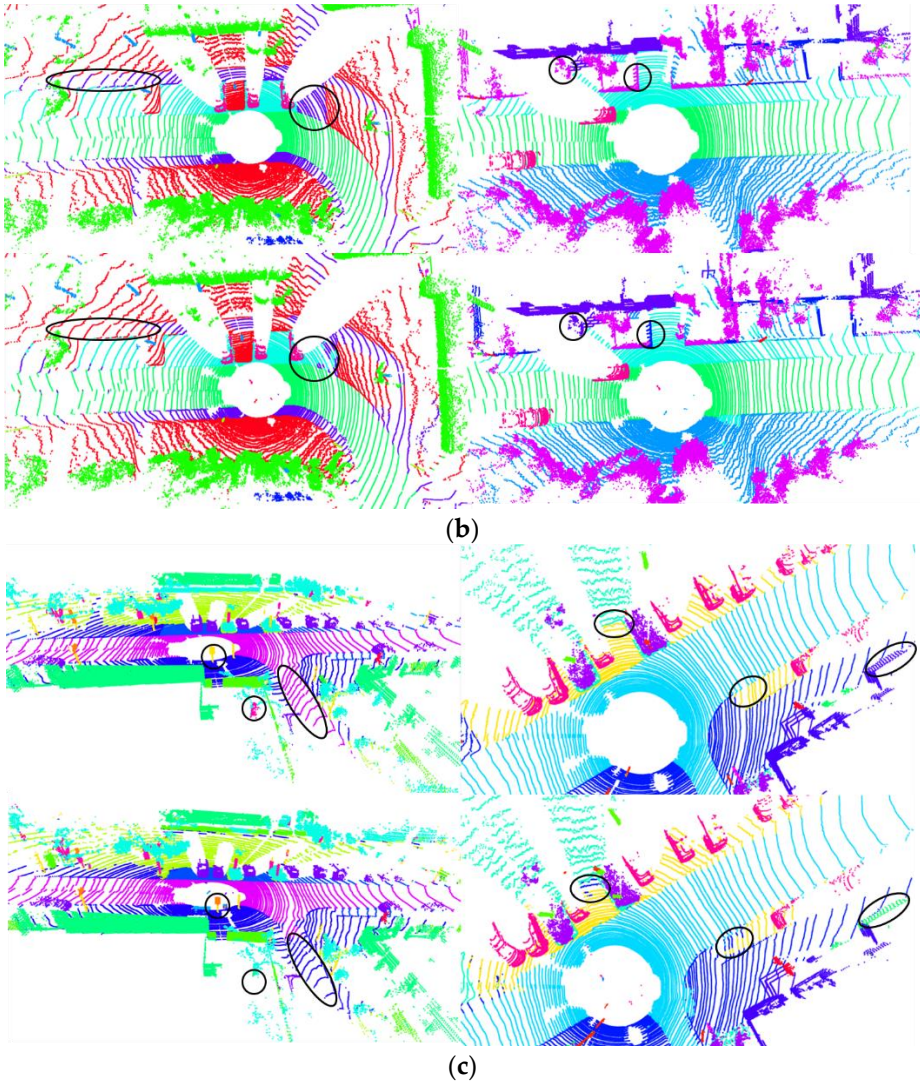


Figure 11. Results of the visualization on SemanticKITTI. First row is the ground truth. Second row shows the prediction results, where an incorrect prediction is circled. (**a**–**c**) Two different scenarios in the SemanticKITTI dataset. The figure shows the prediction results of six different scenarios in total.

4.5. Ablation Experiment

4.5.1. Naïve Local Transformer Structure and Improved Local Transformer Structure

Table 5 shows a comparison of the performances between the naïve local transformer structure and the improved local transformer structure on the S3DIS dataset. In the ablation experiment, we fused the two attention weight matrices by adding them. The experiment results proved the importance of adding the key matrix obtained by the position embedding matrix. Using the improved local transformer structure, OA improved by 0.8%, mAcc improved by 1.6%, and mIoU improved by 1.5%.

Table 5. Segmentation results on area 5 of S3DIS. (Naïve: naïve local transformer; Improved: improved local transformer).

Methods	OA	mAcc	mIoU
Naïve	86.8	70.3	62.6
Without cross-skip selection	87.0	70.1	62.5
Without normal	87.1	68.9	61.6
Improved	**87.6**	**71.9**	**64.1**

The results of the visualization are shown in Figure 12. According to these results (the part circled in the figure), it can be seen that between some objects in which the geometric structures are too similar and cause confusion (such as walls and windows, and walls and doors), the improved local transformer structure (adding the attention weight matrix to explore the geometric relationship) obtains better segmentation results than the naïve local transformer structure.

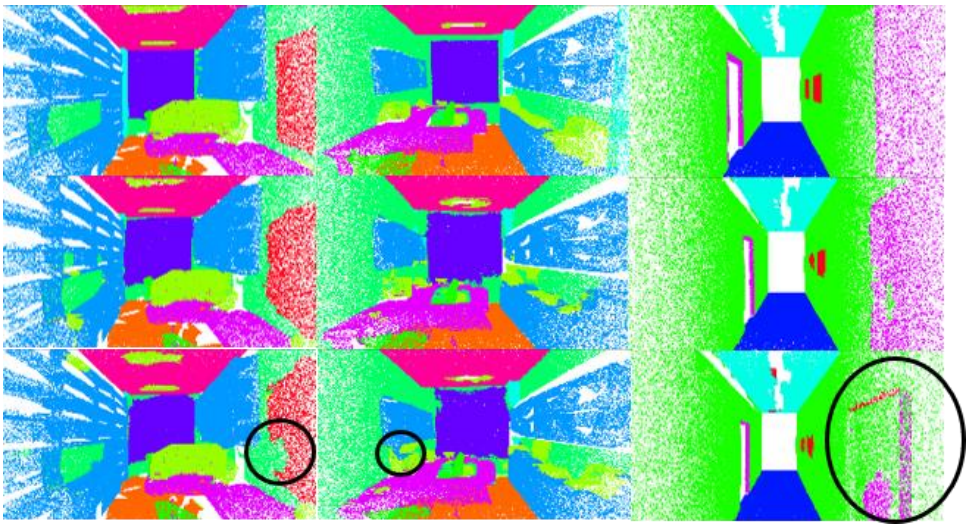


Figure 12. Visualization results. First row is the ground truth. Second row shows the results when using the improved local transformer block. The third row shows the results when using the original local transformer block.

4.5.2. Cross-Skip Selection Method

Our experiments verified the effectiveness of the cross-skip selection of neighbors. The results are shown in Table 5. The OA will increase by 0.6%, mAcc will increase by 1.8%, and mIoU will increase by 1.6%. The results of the visualization are shown in Figure 13. At the junction between objects, the similarity between points is not necessarily positively correlated with the distance between them. Points belonging to different objects may interfere with each other and affect the segmentation performance. Using cross-skip selection of the neighbors can expand the receptive field of each point. The results of the visualization are shown in Figure 14. In the places marked in blue circles, using this method will obtain better segmentation results.

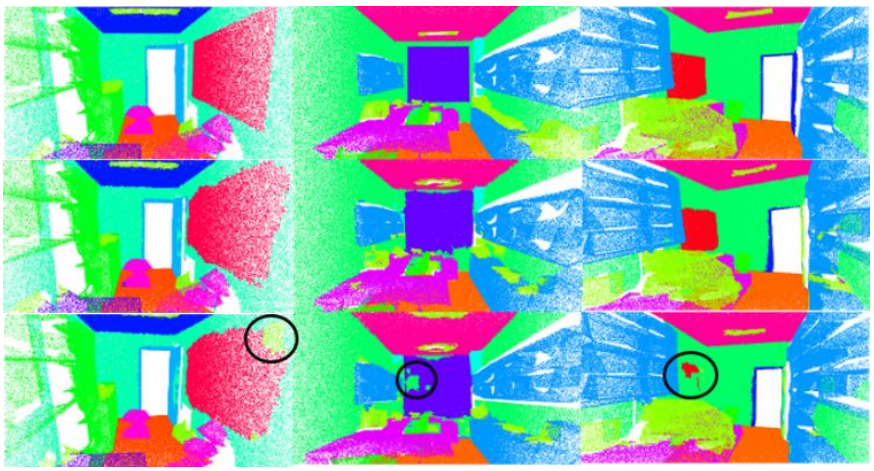


Figure 13. Visualization results. First row is the ground truth. Second row shows the results when using cross-skip selection. The third row shows the results when not using cross-skip selection.

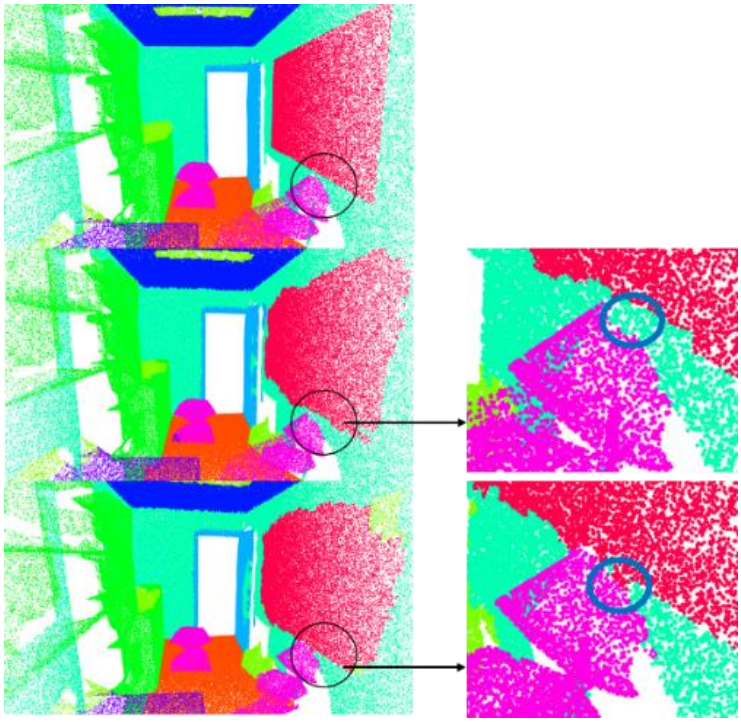


Figure 14. Visualization results. First row is the ground truth. Second row shows the results when using cross-skip selection. The third row shows the results when not using cross-skip selection.

4.5.3. Normal Feature

Table 5 shows a comparison between the results when using and not using normal vector features on the S3DIS dataset. It can be found that, after using normal vector features, OA will be improved by 0.5%, mAcc will be improved by 3%, and mIoU will be improved by 2.5%. This is mainly due to the fact that the normal vectors of most points belonging to the same object are similar or continuously changing.

5. Conclusions

In this paper, we first proposed a muti-scale transformer network for semantic segmentation of a 3D point cloud. This network structure can effectively extract the local and global features of a 3D point cloud. Second, in the local transformer structure, two different attention weight matrices are obtained, with the aim of obtaining the feature similarities and local geometric structure relationships between points. Moreover, we proposed two strategies for fusing the two attention weight matrices. Through ablation experiments, it was proven that the structure can extract the nearest neighbor feature and obtain better segmentation performances between objects with similar geometric structures. Third, we proposed a parallel encoder layer with the cross-skip neighbor selection method, which obtains a larger receptive field for each point without increasing the dimensions of the neighbor feature matrix. From the results of the visualization, it can be seen that this method obtains better results at the junction of multiple objects.

In future work, the following two aspects will be explored. First, this paper proposed two methods for fusing two different attention weight matrices in the local transformer. Whether there is a more effective and efficient fusion method is worthy of further exploration and research. Second, the transformer itself has the disadvantages of requiring a large number of calculations and having low efficiency. The work conducted in this paper was an attempt at applying it to large-scale datasets. However, improving the efficiency and real-time performance without losing accuracy needs further research.

Author Contributions: Conceptualization, Z.W. and Y.W.; methodology, Z.W. and L.A.; software, Z.W. and L.A.; validation, Z.W.; formal analysis, Z.W.; investigation, Z.W.; resources, Z.W., Y.W. and H.L.; data curation, Z.W.; writing—original draft preparation, Z.W.; writing—review and editing, Z.W., Y.W., L.A., H.L. and J.L.; visualization, Z.W.; supervision, H.L. and J.L.; project administration, Y.W. and J.L.; funding acquisition, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by National Natural Science Foundation of China (Grant Nos. 61871376).

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

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

Appendix A

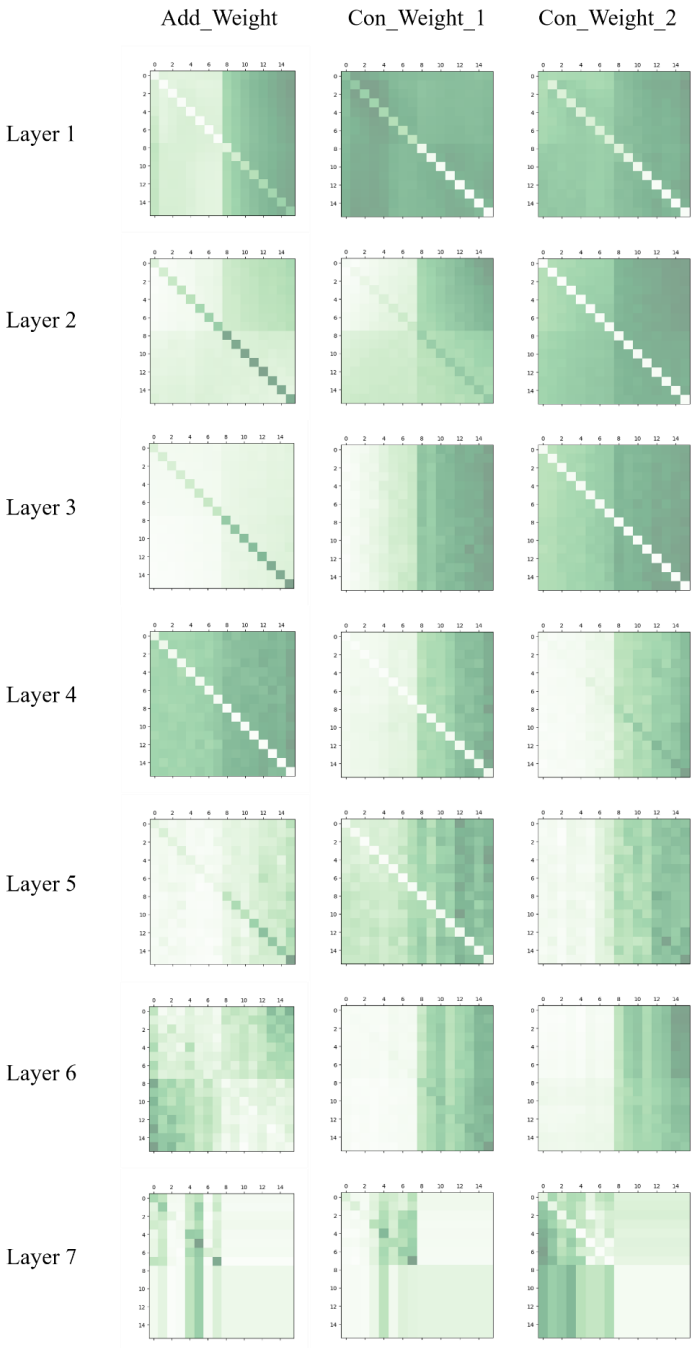


Figure A1. Visualization of the attention weight matrices. The first column is W_{add} obtained using Formula (16). The second and the third columns are W_1 and W_2 when using the second fusion method, which can be obtained using Formulas (13) and (15).

References

1. Tran, L.V.; Lin, H.Y. BiLuNetICP: A Deep Neural Network for Object Semantic Segmentation and 6D Pose Recognition. *IEEE Sens. J.* **2021**, *21*, 11748–11757. [CrossRef]
2. Claudine, B.; Rânik, G.; Raphael, V.C.; Pedro, A.; Vinicius, B.C.; Avelino, F.; Luan, J.; Rodrigo, B.; Thiago, M.P.; Filipe, M.; et al. Self-Driving Cars: A Survey. *Expert Syst. Appl.* **2021**, *165*, 113816.
3. Cortinhal, T.; Tzelepis, G.; Aksoy, E.E. SalsaNext: Fast, Uncertainty-Aware Semantic Segmentation of LiDAR Point Clouds. *arXiv* **2020**, arXiv:2003.03653.
4. 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–18 June 2020; pp. 9598–9607.
5. Rao, Y.; Lu, J.; Zhou, J. Spherical Fractal Convolutional Neural Networks for Point Cloud Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seoul, Korea, 27–28 October 2019; pp. 452–460.
6. Gerdzhev, M.; Razani, R.; Taghavi, E.; Liu, B. TORNADO-Net: MulTiview tOtal vaRiatioN semAntic segmentation with Diamond inception module. In Proceedings of the IEEE international Conference on Robotics and Automation, Xi'an, China, 30 May–5 June 2021; pp. 9543–9549.
7. Zhou, Z.; Zhang, Y.; Foroosh, H. Panoptic-PolarNet: Proposal-Free LIDAR Point Cloud Panoptic Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 13194–13203.
8. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.; Koltun, V. Point Transformer. *arXiv* **2020**, arXiv:2012.09164.
9. Milioto, A.; Vizzo, I.; Behley, J.; Stachniss, C. RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Macau, China, 4–8 November 2019; pp. 4213–4220.
10. 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 International Conference on Robotics and Automation, Orlando, FL, USA, 21–26 May 2018; pp. 1887–1893.
11. Liong, V.E.; Nguyen, T.N.T.; Widjaja, S.; Sharma, D.; Chong, Z.J. AMVNet: Assertion-based Multi-View Fusion Network for LiDAR Semantic Segmentation. *arXiv* **2020**, arXiv:2012.04934.
12. Maturana, D.; Scherer, S. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Hamburg, Germany, 28 September–2 October 2015; pp. 922–928.
13. Graham, B.; Engelcke, M.; Maaten, L.V.D. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 9224–9232.
14. Qi, C.R.; Su, H.; Mo, K. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 77–85.
15. 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 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–7 December 2017; pp. 5105–5114.
16. Wu, W.; Qi, Z.; Li, F. PointConv: Deep Convolutional Networks on 3D Point Clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 9613–9622.
17. Zhao, H.; Jiang, L.; Fu, C. PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 5560–5568.
18. Thomas, H.; Qi, C.R.; Deschaud, J.E.; Marcotegui, B.; Goulette, F.; Guibas, L.J. KPConv: Flexible and Deformable Convolution for Point Clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–3 November 2019; pp. 6410–6419.
19. Hu, Q.; Yang, B.; Xie, L.; Rosa, S.; Guo, Y.; Wang, Z.; Trigoni, A.; Markham, A. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 11105–11114.
20. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transact. Graph.* **2019**, *149*, 1–12. [CrossRef]
21. Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; Shan, J. Graph Attention Convolution for Point Cloud Semantic Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 10288–10297.
22. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16 × 16 Words: Transformers for Image Recognition at Scale. In Proceedings of the International Conference on Learning Representations, Vienna, Austria, 3–7 May 2021.
23. Li, Y.; Zhang, K.; Gao, J. LocalViT: Bringing Locality to Vision Transformers. *arXiv* **2021**, arXiv:2104.05707.

24. Cho, K.; Merrienboer, B.V.; Gülçehre, Ç.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014.
25. Armeni, I.; Sax, S.; Zamir, A.R.; Savarese, S. Joint 2D-3D-semantic data for indoor scene understanding. *arXiv* **2017**, arXiv:1702.01105.
26. Hackel, T.; Savimov, N.; LADICKY, L.; Wegner, J.D.; Schindler, K.; Pollefeys, M. Semantic3D.net: A new Large-scale Point Cloud Classification Benchmark. *arXiv* **2017**, arXiv:1704.03847. [CrossRef]
27. 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 IEEE International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 9297–9307.
28. Tatarchenko, M.; Park, J.; Koltun, V.; Zhou, Q. Tangent Convolutions for Dense Prediction in 3D. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 3887–3896.
29. Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; Chen, B. PointCNN: Convolution On X-Transformed Points. *arXiv* **2018**, arXiv:1801.07791.
30. Landrieu, L.; Simonovsky, M. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4558–4567.
31. Qiu, S.; Anwar, S.; Barnes, N. Semantic Segmentation for Real Point Cloud Scenes via Bilateral Augmentation and Adaptive Fusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 1757–1767.
32. Boulch, A.; Puy, G.; Marlet, R. FKAConv: Feature-Kernel Alignment for Point Cloud Convolution. In Proceedings of the Asian Conference on Computer Vision, Cham, Switzerland, 30 November–4 December 2020.
33. Fan, S.; Dong, Q.; Zhu, F.; Lv, Y.; Ye, P.; Wang, F.Y. SCF-Net: Learning Spatial Contextual Features for Large-Scale Point Cloud Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 14499–14508.
34. Zhang, Z.; Hua, B.S.; Yeung, S.K. ShellNet: Efficient Point Cloud Convolutional Neural Networks using Concentric Shells Statistics. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–3 November 2019; pp. 1607–1616.
35. Truong, G.; Gilani, S.Z.; Islam, S.M.S.; Suter, D. Fast Point Cloud Registration using Semantic Segmentation. In Proceedings of the Digital Image Computing: Techniques and Applications, Perth, Australia, 2–4 December 2019; pp. 1–8.
36. Gong, J.; Xu, J.; Tan, X.; Song, H.; Qu, Y.; Xie, Y.; Ma, L. Omni-supervised Point Cloud Segmentation via Gradual Receptive Field Component Reasoning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 11668–11677.
37. 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 International Conference on Robotics and Automation, Montreal, Canada, 20–24 May 2019; pp. 4376–4382.