Point Attention Network for Semantic Segmentation of 3D Point Clouds

Mingtao Feng[a], Liang Zhang[b], Xuefei Lin[c], Syed Zulqarnain Gilani[d] and Ajmal Mian[d]

[a]Hunan University, Changsha, China; [b]Xidian University, Xi'an, China; [c]Hunan Agricultural University, Changsha, China; [d]The University of Western Australia, Perth, Australia

Abstract

Convolutional Neural Networks (CNNs) have performed extremely well on data represented by regularly arranged grids such as images. However, directly leveraging the classic convolution kernels or parameter sharing mechanisms on sparse 3D point clouds is inefficient due to their irregular and unordered nature. We propose a point attention network that learns rich local shape features and their contextual correlations for 3D point cloud semantic segmentation. Since the geometric distribution of the neighboring points is invariant to the point ordering, we propose a Local Attention-Edge Convolution (LAE-Conv) to construct a local graph based on the neighborhood points searched in multi-directions. We assign attention coefficients to each edge and then aggregate the point features as a weighted sum of its neighbors. The learned LAE-Conv layer features are then given to a point-wise spatial attention module to generate an interdependency matrix of all points regardless of their distances, which captures long-range spatial contextual features contributing to more precise semantic information. The proposed point attention network consists of an encoder and decoder which, together with the LAE-Conv layers and the point-wise spatial attention modules, make it an end-to-end trainable network for predicting dense labels for 3D point cloud segmentation. Experiments on challenging benchmarks of 3D point clouds show that our algorithm can perform at par or better than the existing state of the art methods.

Keywords:
Semantic segmentation, 3D point cloud, point attention network, deep learning

1. Introduction

With the widespread availability of 3D scanning devices and depth sensors [1], 3D geometric data is being increasingly used in many different application domains such as robotics, autonomous driving, 3D scene understanding, city planning, infrastructure maintenance etc [2, 3, 4, 5]. Several representations of 3D shape have been investigated, such as depth maps, voxels, multi-views, meshes and point clouds [6]. However, point cloud is arguably the simplest format for 3D data representation and has hence attracted increasing research interest. Similar to the pixels in a 2D image, points in the three-dimensional coordinate system are basic building units of point clouds, which naturally encode the geometric features and their spatial distributions of a real 3D scene.

The extraction of meaningful information from 3D point clouds requires semantic segmentation. Point cloud semantic segmentation has been a challenging and active research topic for the last few years. Unlike pixels of 2D images which have a rectangular grid-like structure with no missing bits, 3D point clouds are sparse, irregular, unordered and with missing regions due to the limited range of scanners and occlusions. While deep learning has been very successful in semantic segmentation of 2D images, its use for 3D point clouds has not been fully exploited yet. Qi et al. [7] first proposed PointNet that learns point features directly from unordered point sets. In PointNet, all 3D points are independently passed through a set of multi-layer perceptions (MLP) and then aggregated to a global feature using max-pooling. Recent research directions focus on extending the basic idea of PointNet to incorporate local geometric features for abstracting more discriminative high level features [8, 9, 10]. Among these methods, Pointnet++ [8] exploited neighborhood points within a ball query radius, where each local point is processed separately by a PointNet-based hierarchical network. However, the relationships between local points are neglected. Recently dynamic graph CNN [9] was proposed which considers neighborhood points as a local graph and uses a filter generating network to assign edge labels. Since the edge-conditioned network does not consider the order of local points, it does not have transformation invariance. Similar to dynamic graph CNN [9], dynamic edge conditioned filters [10] were introduced as an edge function to encode local information by combining the relative coordinates (raw features) between the center point and its K-nearest neighbors (KNN). Although dynamic edge conditioned filters [10] attempt to use a function designed to handle local points, it does not fully exploit the geometrical correlations of the local neighborhood points.

To address the above short comings, we propose a local attention-edge convo-

lution (LEA-Conv) layer that extends the ideas of [8, 9] and [10]. The LAE-Conv layer constructs a local graph based on the neighborhood points searched along multiple directions. Unlike KNN and ball query methods, we propose a multi-directional search strategy that finds all neighborhood points from 16 directions spread systematically within a ball query making the local geometric shape more generalizable across space. After the search operation, LAE-Conv layer assigns attention coefficients to each edge and then aggregates the central point features as a weighted sum of its neighbors. Aggregating features from a group of points with their contribution coefficients, rather than a single max-pooling operation, better exploits the correlations between points to get accurate and robust local geometric details. Moreover, LAE-Conv layer is invariant to the ordering of points and can implicitly infer how the points contribute to the overall 3D shape.

Equipped with the LAE-Conv layer, we are able to design hierarchical deep learning architectures on point clouds for semantic segmentation. Since each LAE-Conv layer has a limited local receptive field, each unit of the output features (at the initial layers) exploits correlations within its local scale only. However, later LAE-Conv layers have progressively larger receptive fields enabling the network to learn hierarchical features. While existing networks [8, 11, 9] capture multi-scale shapes for high-level point feature learning, they do not leverage the long-range contextual relationship among points belonging to the same categories, which is important for semantic segmentation. Superpoint graphs [12] employed a recurrent neural network to exploit long-range dependencies based on an unsupervised geometric partitioning. However, that method relies heavily on the partitioning results. To address the above problems, in this paper, we propose a point-wise spatial attention module, which captures long-range contextual information in the spatial dimension. Features obtained from LAE-Conv layer are fed into the point-wise spatial attention module to generate a global dependency matrix which models the correlations between any two points of the feature maps. Through multiplying the dependency matrix with original features, the differences between point features of the same category are reduced. Hence, any two points with similar features can contribute mutual improvement regardless of their spatial distance.

Using the proposed LAE-Conv layer and point-wise spatial attention model as the main building blocks, we design a U-shape network to predict the dense labels for semantic segmentation of 3D point clouds. The unorganized 3D points (raw data) are input directly to our point attention network comprising an encoder and a decoder. This is different from other approaches [8, 11, 9] since our method stacks the point-wise attention module after the LAE-Conv layer at different stages of the

network enabling it to learn more accurate local geometric features and long range relationships.

To summarize, our contributions include: (1) A novel local attention-edge convolution (LAE-Conv) layer to encode point features using a weighted sum of its neighborhood points with edge attention coefficients. The proposed multi-directional search strategy makes the local geometric shape more generalizable across space. (2) A novel point-wise spatial attention module that learns the long-range contextual information and significantly improves the segmentation results by boosting the representation power of local features obtained from the LAE-Conv layers. (3) Extending the U-shaped network to incorporate the proposed LAE-Conv layer and point-wise spatial attention module. Experimental results show that our method obtains on pair or better performance than existing state-of-the-art methods quantitatively and qualitatively on challenging benchmark datasets. Finally, we show that our proposed point attention block can generalize to other networks and improve their performance.

2. Related Work

A number of deep learning architectures have been recently proposed to learn directly from 3D point cloud data or its derived representations for applications such as semantic segmentation, object part segmentation and object categorization. We provide a brief survey of these methods and divide them into three categories based on the underlying data representations they use.

2.1. Indirect methods

This category includes methods that transform the irregular 3D point cloud data to a canonical form so that traditional convolutions can be applied [13, 14]. Volumetric representations [15, 6, 16, 17, 18] are the most common canonical form used by these methods due to their simplicity. However, voxel representations have cubic complexity leading to dramatic increase in the memory consumption and computing resources required to process even medium size point clouds. To alleviate this problem, Octree-Net [19, 20] and Kd-Net [21] have been proposed which skip representation and computations at empty spaces to save memory and processing resources respectively [22]. Moreover, sparse convolutional operations, where the activations are kept sparse in the convolution layers [23, 24], have been introduced to process spatially-sparse 3D point clouds. Nevertheless, the kernels are still dense and inefficient in their implementation. Multi-view convolutional neural networks and their variants [25, 26, 27, 28] have also

been proposed. These methods render the 3D shape from multiple pre-defined views, which are then processed by conventional image-based convolution networks. The main drawback of the multi-view frameworks is that the 3D geometric information is not always fully retained in the 2D projections.

The sparse lattice networks proposed by Hang et al. [29] project the input 3D points onto a high dimensional lattice, perform standard spatial convolution on it and then filter the features back to the input points. Matan et al. [30] extended the function over point cloud to a volumetric function, where volumetric convolution is applied and then a restriction operator is used to do the inverse action. Qiangui et al. [31] used a slice pooling layer to project unordered point clouds into an ordered format, making it feasible to apply traditional deep learning algorithms. Fully convolutional networks [32] have been proposed that sample the input point cloud uniformly and use PointNet as a low-level feature learner, followed by 3D convolutions to learn features at multiple scales. Finally, tangent convolutions [33] have also been proposed that operate directly on surface geometry in the tangent space. Although the above methods have used deep learning techniques to realize the 3D data analysis tasks, they have not used the 3D point clouds directly. We believe that learning directly from raw 3D point cloud data can achieve higher accuracy and efficiency as learning from raw data is the major strength of deep learning.

2.2. Graph convolution methods

Graph convolutional methods combine the power of convolution operation with graph representations of irregular data. Graph convolutional networks have been designed to perform convolutions either in the spectral or spatial domain. More recently, Joan et al. [34] proposed a generalization of convolution for graph via the Laplacian operator. In that method, the spectral network can learn convolutional layers with a number of parameters for low dimensional graphs. Wang et al. [35] proposed a local spectral graph convolution to construct local graph from a point's neighborhood and aggregate information from nodes using their spectral coordinates. The PointNet++ architecture is then applied along with the local spectral graph convolution layers and graph pooling layers. The regularized graph convolution network proposed by Gusi et al. [36] treats point cloud as a graph and defines convolution operation over it. Moreover, a graph smoothness prior is used in the loss function to regularize the learning process. Graph Laplacian based methods have a number of drawbacks including the computational complexity of Laplacian eigen-decomposition, the large number of parameters to express the convolutional filters, and the lack of spatial localization. Different from these

methods, Martin et al. [10] proposed a convolution-like operation on graph signals in the spatial domain and used an asymmetric edge function to describe the relationships between local points. However, the edge labels are dynamically generated and hence, the irregular distribution of local points is not taken into account. This method was improved by Wang et al. [9] through max pooling operation on local features. However, max pooling operation is still unable to fully utilize the correlations of local points. Our proposed method exploits local feature learning using a completely different approach. We propose a local attention-edge convolution layer that learns local relationships between points.

2.3. Point cloud methods

Many researchers have proposed deep learning architectures that learn directly from point clouds. One of the earliest methods in this category is the PointNet [7] that operates on point clouds using multi-layer perception (MLP). PointNet is robust to the global transformation of 3D shape because the spatial transformer network [37] is used to learn the 3D alignment. The main limitation of PointNet is that it only relies on the max-pooling layer to learn global features. Since PointNet does not consider local relationships, Qi et al. [8] introduced an improved network named PointNet++, which exploits local geometric features in point sets and aggregates them for hierarchical inference. However, PointNet++ still treats points within local regions individually and does not consider relationships between the neighborhood points.

Later, Francis et al. [38] designed a multi-scale architecture to enlarge the receptive field over the 3D scene by incorporating larger-scale spatial grid blocks into PointNet. Loic et al. [12] used an unsupervised method to cluster input points into superpoint graphs, then fed the graphs to PointNet-based gated recurrent unit. Li et al. [11] proposed X-Conv layer instead of MLP to permute unordered local points into a latent potentially canonical order. A similar approach was proposed in [39], where kernel correlation was introduced to incorporate local information extracted from point cloud by PointNet. Wang et al. [40] introduced a similarity group proposal network for point cloud instance segmentation, which use a similarity matrix to produce a grouping proposal based features extracted from PointNet. Different from these PointNet-based frameworks, Hua et al. [41] presented a point-wise convolution operator that can be applied to each point of the point set. Recently, Zhao et al. [42] proposed PointWeb for point cloud processing, which connects all points densely in a local neighborhood for better encoding local geometric features. Wu et al. [43] introduced PointConv, a nonlinear function kernel for point cloud, which is used to learn the translation-invariant and

Algorithm 1 LAE-Conv Operation

Input: Input local points h, central point p_i; Number of selected points m in each bin;

Output: Filtered central point $p_{i_{LAE}}$;

1: Search K neighbor points p_j of p_i in the point cloud h;
2: $p_j - p_i$: move points p_j to local coordinate system of p_i;
3: $W(p_j - p_i)$: transform the input points into higher-level features ;
4: α_{ij}: compute normalized attention edge coefficients with softmax;
5: p_i': use graph attention aggregator to obtain updated feature at p_i;
6: MLP(p_i'): feature transformation operation;
7: **return** $p_{i_{LAE}}$;

permutation-invariant features in 3D space. Wang et al. [44] designed a graph attention kernel to adapt to the local geometric, which is useful for fine-grained segmentation.

A common limitation of all the aforementioned methods is that they are unable to simultaneously exploit fine local details and long-range contextual information. We fill this gap and propose a network that learns local geometrical features using their edge attention coefficients and allows deep learning architectures to exploit fine details as well as interactions over longer distances.

3. Proposed Approach

We first give details of the LAE-Conv layer that captures accurate local geometric details. Next, we explain the point wise spatial attention module that aggregates the long-range contextual information based on the output of LAE-Conv layers. Finally, we present a general framework of our network.

3.1. Local Attention-Edge Convolution (LAE-Conv)

The Local Attention-Edge Convolution (LAE-Conv) layer forms the basic component of our point attention network architecture for 3D point cloud semantic segmentation. Inspired by DGCNN [9], ECC [10], GATs [45] and Non-local network [46], we construct a multi-directional neighborhood graph and apply graph attention mechanism to compute local edge features. Similar to traditional convolution in images, LAE-Conv explores local regions to leverage correlations between unordered points and exploits the local geometric structure of the points. We summarize the LAE-Conv operator in Algorithm 1.

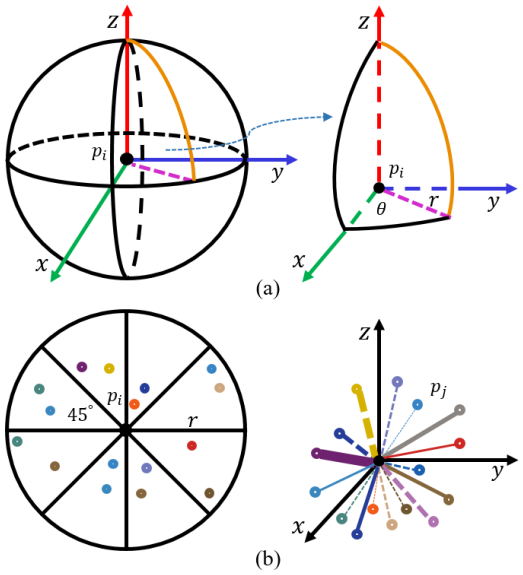

Figure 1: (a) An illustration of our multi-directional search method. The ball space around the center point within the search radius is divided into 16 uniform directions. The azimuth θ, radius r and number m of selected points in one cube are hyperparameters. (b) When $m = 1$, 16 neighborhood points are considered along different directions. If all neighbors are projected onto the xy coordinate plane, we can see that there are two points in each of the eight directions. The thickness of the line connecting the center point to the neighbors represents different contributing values.

3.1.1. Multi-directional Search

In image convolution operation, the local region of a pixel can be represented in a grid-like structure given a convolution kernel size. However, the neighborhood of a center point (in a point cloud) is defined by metric distance in a 3D coordinate system where neighboring points are irregularly distributed. To robustly leverage local point correlations, we endeavour to explicitly capture geometric information in different orientations. Given an unordered point cloud $P = \{p_1, p_2, \ldots, p_N\}$ with $p_i \in \mathbb{R}^C$, where N is the number of points, and C is the feature dimension at each point. When each point is represented by its 3D coordinates $p_i = (x_i, y_i, z_i)$, then $C = 3$. We denote a central point in P as p_i, and its K neighbors in P as p_j, $j \in \mathcal{N}(i)$. As shown in Figure 1(a), the space around the reference point p_i within a radius of r is split into 16 bins, where each bin indicates a direction. Each bin has an azimuth angle $\theta = \angle 45$. Within the spatial range represented by each bin, we select m nearest points of p_i from all the points that fall in that bin and use their features to represent the bin, i.e. when

$m = 1, 2, 3...$, $K = 16, 32, 48....$ Since some points far away from p_i are not very useful to represent p_i, we set the radius r empirically as a hyper-parameter according to each layer. In case there are insufficient points inside a bin, point p_i is repeated. This is similar to self convolution.

Two common ways for range query are K-nearest neighbor (KNN) search and ball query. KNN returns a fixed number of K neighboring points while ball query returns all points that are within a radius. The local shape will not be well represented if all selected points, using either of the methods, are from a small region or one direction. Different from KNN and ball query, our search method guarantees that neighborhood points are from different directions to ensure sufficient expressive power of encoding the local geometric information. We compare the effectiveness of our search method over ball query and KNN in the experiments section.

3.1.2. Aggregation

For a set of local points $h = \{p_i, p_{j_1}, p_{j_2}, \cdots p_{j_K}\}$, $h \in \mathbb{R}^C$, where p_i is the central point and others are its K neighbors, we consider a graph $G = (V, E)$, where V is a finite set of points with $|V| = K + 1$ and $E \subseteq V \times V$ is a set of directed edges $\{(p_i, p_{j_1}), (p_i, p_{j_2}) \cdots (p_i, p_{j_K})\}$. We define the attention edge coefficients as e_{ij}, which represent the importance of neighbors p_j to the central point p_i, computed by an attention mechanism a.

$$e_{ij} = a(Wh_i, Wh_j) \tag{1}$$

Where $W \in \mathbb{R}^{C \times C'}$ is a learnable weight matrix that transforms the input point set to higher-level features, h_i and h_j represent the central point and its neighbors respectively and the mechanism a is a single layer MLP, parametrized by a weight vector $\vec{\mathrm{a}} \in \mathbb{R}^{C'}$. To make the edge coefficients easily comparable across different points, we use the softmax function to normalize them across all neighbors of the reference point p_i:

$$\alpha = \text{softmax}(e_{ij}) = \frac{\exp\left(e_{ij}\right)}{\sum_{j \in \mathcal{N}(i)} \exp\left(e_{ij}\right)}. \tag{2}$$

The final edge coefficients computed by the attention mechanism may then be expressed as:

$$\alpha_{ij} = \frac{\exp\left(a(W(p_j - p_i))\right)}{\sum_{j \in \mathcal{N}(i)} \exp\left(a(W(p_j - p_i))\right)}, \tag{3}$$

Where the neighbor points of the central point are transformed to local coordinate systems by $(p_i - p_j)$ and then the local coordinates of each point are lifted to higher-order features by W.

Once obtained, the normalized edge coefficients α_{ij} are used to assign attributes to each edge. Our approach computes the filtered feature at point p_i as a weighted sum of points in its neighborhood. The proposed commutative aggregation method not only solves the problem of undefined point ordering, but also smoothes out the structural information. The local graph attention aggregator is defined as

$$p_i' = \sum_{j \in \mathcal{N}_{p_i}} \alpha_{ij} W p_j, \tag{4}$$

where p_i' is the updated features of central point p_i.

3.1.3. Transformation

Now we have an aggregated representation for the central point p_i. It is natural to add a feature transformation function f to incorporate additional non-linearity and increase the learning capacity of the model. The transformation can be realized by MLP with a non-linear activation function. The output of the transformation function is $p_{i_{LAE}} : 1 \times C'$. The proposed LAE-Conv layer is described in Algorithm 1.

3.2. Point-wise Spatial Attention Block

The output point cloud P_{LAE}: $N \times C'$ of the LAE-Conv layer have rich representation power for local geometric features. However, since each LAE-Conv layer have a local receptive field, individual units of the filtered features are unable to exploit contextual information outside of their local regions. In P_{LAE}, features corresponding to the points with the same label are significantly different when the points are far apart. These differences affect the point wise segmentation accuracy of the scene as a whole. To address this issue, we focus on the global spatial relationships to boost the representation power of the LAE-Conv layer. We design a point-wise spatial attention module that captures the global dependencies by building associations among features within the point set. We demonstrate that by stacking these blocks after LAE-Conv layers, we can construct local-global architectures that adaptively encode long-range contextual information, thus improving the semantic segmentation accuracy of 3D point clouds that cover large areas. Next, we introduce a process to adaptively aggregate point-wise spatial contexts.

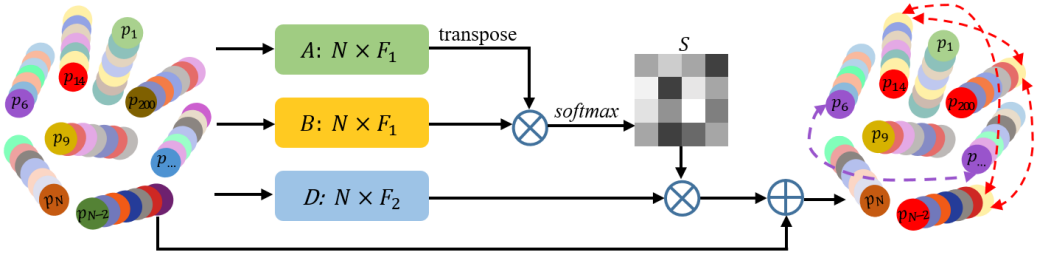

Figure 2: The proposed point-wise spatial attention module. The feature maps are represented by the shape of their tensors, e.g., $N \times F_1$ where N denotes the number of points and F_1 denotes the feature dimension. For simplicity, we set the batch size to 1. $\otimes$ denotes matrix multiplication, and $\oplus$ denotes point-wise sum. The green, yellow and blue boxes denote MLP layers. Long-range correlations are learned once the input features pass through this module.

Inspired by the position attention operation [47], we define a point-wise spatial attention module for 3D point clouds. As illustrated in Figure 2, two MLP layers are used to transform the local feature P_{LAE} into two new representations A and B respectively, where $A, B \in \mathbb{R}^{F_1}$. We compute relationships between different points based on the transpose of A and B. Unlike [47], we calculate the spatial correlations of all points directly from the transpose of A and B without reshaping the matrices, hence, maintaining the original space distribution. Softmax is then applied to normalize relationship map to get the point-wise spatial attention map S with size $N \times N$:

$$s_{ij} = softmax\left(\frac{exp(A_i \cdot B_j)}{\sum_{i=1}^{N} exp(A_i \cdot B_j)}\right), \tag{5}$$

where i and j denote the point positions in A and B respectively, s_{ij} is the i^{th} point's impact on the j^{th} point, and $\cdot$ denotes matrix multiplication. We show that two points have a strong correlation when their features have similar semantic information.

At the same time, the local feature P_{LAE} is transformed to a new feature $D \in \mathbb{R}^{F_2}$ by an MLP layer. This is followed by a matrix multiplication between S and D. Finally, the output is multiplied by a scale parameter α and element-wise summation is performed with the features P_{LAE} to obtain the final output $P_{final} \in \mathbb{R}^{N \times C''}$ as follows:

$$P_{final} = S \cdot D + P_{LAE}, \tag{6}$$

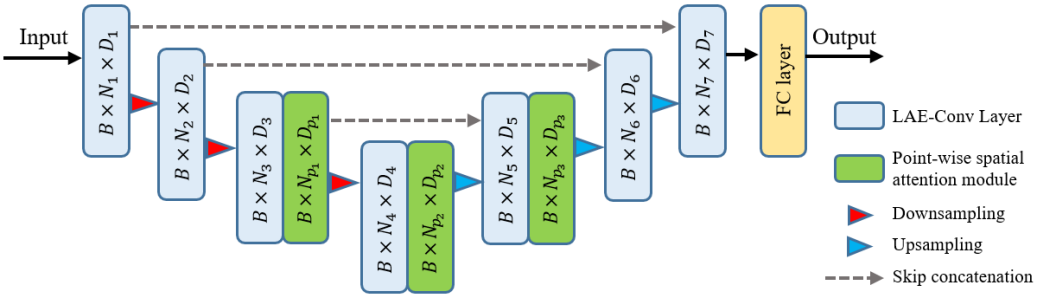

Figure 3: Illustration of the proposed point attention network for point cloud segmentation. The encoder and decoder parts are based on the LAE-Conv layer and point-wise spatial attention module. B, Ni and Ci denote the batch size, point number and point feature dimension respectively. The downsampling and upsampling processes are followed by [8]. The encoder and decoder parts are linked by three skip connections.

where $\cdot$ denotes matrix multiplication. Here, the resulting feature P_{final} contains a long-range contextual information and selectively aggregates contexts according to the point-wise spatial attention map S. This module improves the feature representation power and is more accurate for 3D point cloud semantic segmentation.

3.3. Network Architecture

For dense point label prediction, the output resolution is high. Moreover, there are multiple objects with different scales in one scene. Selecting the most representative scale for each kind of object is important for semantic segmentation. Following the hierarchical structure of PointNet++ [8], our network consists of encoder and decoder parts. As shown in Figure 3, our point attention network comprises the LAE-Conv layers and point-wise spatial attention modules. At the encoder part, the input point set is processed by three LAE-Conv layers, which transform it into fewer representation points but with richer features. The input point cloud is represented by its 3D coordinates and sometimes with the RGB color values as well. The point-wise spatial attention modules are stacked after the third and fourth LAE-Conv layers to aggregate long range point-wise contextual information from output of the previous LAE-Conv layer. The long-range contextual features along with the local features from LAE-Conv layers together achieve robust and accurate 3D point cloud semantic segmentation.

At the decoder part, three skip connections are used to combine features from the encoders. The point-wise spatial attention module is also inserted after the fifth LAE-Conv layer at the decoder part. In our hierarchical architecture, we use three steps of down-sampling operations and tree steps of up-sampling opera-

tions which are followed by set abstraction and feature propagation modules as in PointNet++ [8]. Finally, all the features in the last decoder layer go through fully connected layer and convert to class probabilities.

4. Comparison with Existing Methods

Our point attention network is a more generalized form of the classic approach PointNet++ [8]. We explain how PointNet++ is a special case of our network. PointNet++ is an extension of [7] with considers local point structure. Given a reference point p_i, ball query search K local points with data size $N_l \times K \times C_l$, PointNet processes the local region points individually and then max pools them to get the most representative point feature as the output $N_l \times C_l'$ of the local region. Different from PointNet++, the LAE-Conv layer constructs the local graph for the K neighbors and central point p_i. We compute attention edge coefficients $e_{ij} = e_{i1}, e_{i2}, \cdots e_{iK}$ to indicate different contributions of each neighbor to the central point. When $e_{ij} = \{e_{i1} = 1 | e_{i2}, e_{i2}, \cdots e_{iK} = 0\}$, p_1 features are selected to represent the local region. We can observe that the basic convolution layer of PointNet++ is an instance of our LAE-Conv layer.

DGCNN [9] uses KNN to establish local point shape and proposes an aggregation operation $max(MLP(p_i, p_j - p_i))$. In that operation, the neighbor points are moved to the local coordinate system first and then stacked with the central point. All the neighbors have equal contribution to the central point, which is equivalent to our operator when all edge coefficients are equal to 1. Since DGCNN is based on PointNet, the receptive field remains constant (K) at different layers, which is a disadvantage when encoding point clouds with different spatial distribution densities.

Similar to PointNet++, PointCNN [11] follows the encoder-decoder architecture and learns a $\mathcal{X}$ transformation to lift the input irregular points into an unknown canonical format, then applying a typical convolution on the transformed point cloud. In PointCNN, the dilated convolution process from image convolution networks is employed to expand the local receptive field of different layers. The local receptive field changes the number of neighborhood points K by adjusting the dilation ratio. Different from the grid structure of local pixels, points are disordered in a three-dimensional coordinate system and the density distribution is not uniform. Although KNN searches for neighborhood points which is controlled by the dilation ratio proportionally, the global geometric features learned by the change of receptive field is limited. To address this issue, our point attention network inserts a point-wise attention module in the high level feature layer.

A crucial difference between these two operations is that the latter assumes a long range dependency, which reduces the gap between features corresponding to the points with the same label encoding more accurate global information. The more similar are the feature representations of the two points, the greater is the correlation between them.

5. Experiments and Discussion

We evaluate the performance of the proposed network on the ShapeNet [48] 3D part segmentation dataset and the two largest point cloud segmentation benchmarks, ScanNet [49] and Stanford Large-Scale 3D Indoor Spaces (S3DIS) [50]. While ShapeNet is synthetic data, ScanNet and S3DIS are real point clouds obtained with a scanner. We perform ablation studies of different design choices and network variations as well as compare the performance of our network with existing state of the art.

5.1. ScanNet

ScanNet [49] contains 1513 scans annotated with semantic voxel labels from 21 categories (bed, refrigerator, floor, table etc. plus other furniture). ScanNet is divided into 1201 training and 312 test samples. Similar to [8], we split the ScanNet training scenes into 2m by 2m by 3m blocks, with 0.5m padding in each direction (x,y,z) and sample 8192 points randomly from each block on the fly. To predict semantic label of every point of the test scene, we similarly split it into similar cubes using a sliding window strategy along the xy plane with different stride sizes. If the same point gets different predictions in the overlap regions, we choose the one with highest confidence.

Although ScanNet also contains RGB values for each point, we only use the xyz coordinates as point features for a fair comparison with other methods. Hence, the input data size for the network is 8192×3. As shown in Figure 3, we use downsampling and upsampling operations from PointNet++ [8] for both the encoder and decoder parts. The output point numbers and feature dimensions of different LAE-Conv layers are $(N_1 = 8192, C_1 = 64)$, $(N_2 = 2048, C_2 = 128)$, $(N_3 = 512, C_3 = 256)$, $(N_4 = 128, C_4 = 512)$, $(N_5 = 512, C_5 = 256)$, $(N_6 = 2048, C_6 = 256)$ and $(N_7 = 8192, C_7 = 128)$ respectively. The fully connected layer with size $(N_{fc} = 8192, C_{fc} = 21)$ converts the final features into class probabilities. We set $(m = 1, K = 16)$ for the neighborhood search. For the three point-wise attention block, the output point numbers and feature dimensions are $(N_{p_1} = 512, C_{p_1} = 256)$, $(N_{p_2} = 128, C_{p_2} = 512)$ and $(N_{p_3} = 512, C_{p_3} = 256)$

Table 1: 3D point cloud semantic segmentation results on ScanNet scenes. The metrics are mean per-class Intersection over Union (mIoU,%) and per voxel overall accuracy (OA, %).

Method	mean IoU	Overall Accuracy (OA)
PointNet [7]	-	73.9
PointNet++ [8]	-	84.5
RSNet [31]	39.35	-
TCDP [33]	40.9	80.9
FCPN [32]	-	82.6
3DRCNN [51]		76.5
PointCNN [11]	-	85.1
Ours	**42.1**	**86.7**

Table 2: Model size and inference time comparison, where "M" means million and "s" denotes second. We use the model file (.cptk) size obtained by the training using tensorflow to represent the complexity of different methods. The entire scenes was tested 5 times and the average time was recorded.

Methods	Size (M)	Time (s)
PointNet [7]	321.9	2.16
PointNet++(msg) [8]	177.3	3.8
DGCNN [9]	180.1	3.94
SpiderCNN(3-layers) [14]	349.5	4.3
Ours	183	3.97

respectively. The initial learning rate is 0.001, batch size is 22 and the momentum is 0.9. We set the decay rate of 0.7 and stop training after 1000 epochs.

Table 1 shows quantitative comparison of our proposed point attention network with PointNet++ [8], PointCNN [11] on the ScanNet dataset. This comparison is done using two metrics, namely the mean per-class IoU (mIoU, %) and per voxel overall accuracy (OA, %). For a fair comparison, Table 1 shows results of baseline methods reported in the original papers since the trained models are not available for testing. Compared to the baseline methods, our network achieves the highest accuracy on both metrics. Table 2 reports the model size and average inference time of a few representative methods [7], [8], [9], [14], where the released source codes are easy to use. Experiments are conducted by a single NVIDIA GTX TitanX GPU with tensorflow and an Intel i7-9700K@3.6

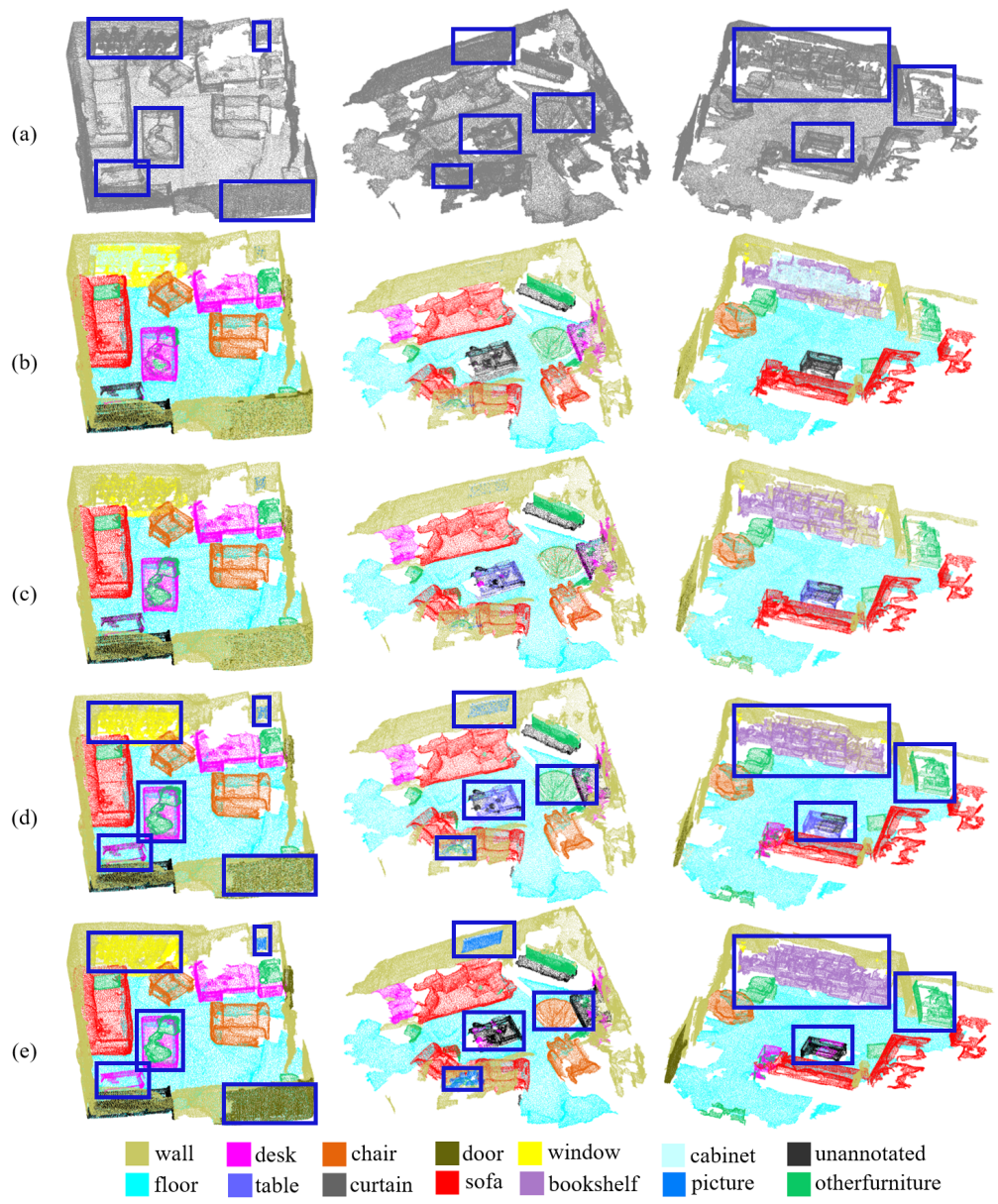

Figure 4: Qualitative comparison on three scenes from ScanNet. (a) Input point cloud with only xyz coordinate features. Semantic segmentation by (b) PointNet++ [8], (c) PointCNN [11] and (d) our method. (e) Ground truth Semantic labels. Colors denote different categories. These scenes contain only 13 categories out of 21. The areas marked by the boxes are some examples where our method performed significantly better than others.

GHZ 8 cores CPU. Compared with these methods, we can see that our proposed architecture improves segmentation results with only marginal extra computation cost.

Figure 4 qualitatively compares the semantic segmentation obtained by PointNet++, PointCNN and our method. We use boxes to highlight some examples where our method performed significantly better than the competitors. In the first scene, the window and the door are embedded in the wall whereas the picture is hung on the wall making the semantic segmentation a real challenge. Our method's output is more regular than that of PointNet++ and PointCNN. The table in the lower left corner is incomplete with a mere skeleton. Hence, segmentation methods like PointNet++ and PointCNN get worse results compared to our method. In the second scene, all the methods get incorrect predictions on the chair that is close to the floor as well as the irregularly shaped desks. This is because the per class samples in ScanNet dataset are unbalanced [49] making existing segmentation methods fail on the rare categories. In the third scene, our method performs better on bookshelves than others. In addition, the un-annotated object in the center of scene is misidentified as table by PointCNN and our method because its shape is more like a table than an ordinary chair.

To better understand the influence of various design choices made in our network, we analysis them on ScanNet.

5.1.1. Ablation study on parameters of LAE-Conv layer

As mentioned in Sec 3.1, there are three options (KNN, ball query and our multi-direction searching method) for searching the neighbors of the central point. We use ScanNet as a test benchmark to compare these options. We also set different point numbers at each cube for our proposed search method. In Table 3, we can see that our method is more efficient for selecting local point shapes. When $(m = 2, K = 32)$ and $(m = 3, K = 48)$, the segmentation accuracy is greatly reduced. This is because the parameters of LAE-Conv layer will increase as the number of neighbors increase. Too many neighbors bring information redundancy, which reduces the efficiency and accuracy of the LAE-Conv layer.

5.1.2. Ablation study on point-wise spatial attention block

To take full advantage of the point-wise spatial attention block, we show the segmentation results with more attention blocks in the network architecture. We add 7 attention blocks (after LAE-Conv layer $1 - 7$), 5 attention blocks (after LAE-Conv layer $2 - 6$) and 3 attention blocks (after LAE-Conv layer $3 - 6$). As shown in the first part of Table 4, more point-wise spatial attention blocks do not

Table 3: Ablation analysis on ScanNet with different search methods and numbers. All results are based on our LAE-Conv layer while other settings are kept constant.

Neighborhood Search Method	Overall Accuracy (OA %)
KNN (K=16)	85.0
Ball query (K=16)	85.3
Proposed Multi-direction (m=1,K=16)	**86.7**
Proposed Multi-direction (m=2,K=32)	85.9
Proposed Multi-direction (m=3,K=48)	84.4

lead to an improvement in performance. One explanation is that more attention blocks massively increase the number of parameters and the network can not find a local optimal solution within the specified training steps on ScanNet. The second part of Table 4 compares same number of attention blocks added to different stages of network. The attention block is added to the right, after the LAE-Conv layer (2,4,6) and (1,4,7) respectively. We can see that the results deteriorate when the attention blocks are added to layers with lower feature dimensions. A possible explanation is that the point features do not contain enough representative semantic information when their dimensions are low, the features of the points with the same labels are significantly different, and the number of parameters of attention block will also increase from (2,4,6) to (1,4,7). Under this condition, the effectiveness of attention block is limited. Finally, we choose to add three attention blocks to the right after LAE-Conv layers (3,4,5) in Figure 3. We also tested adding three attention blocks to vanilla PointNet++ (without MSG and DP [8]) at the corresponding stages as in our network. As shown in the third part of Table 4, the performance of baseline network (vanilla PointNet++ [8]) is improved by 3.4%. This shows that our proposed point attention block is generic and is able to improve the performance of any network architecture.

5.2. S3DIS

The Stanford Large-Scale 3D Indoor Spaces (S3DIS) dataset [50] contains 3D scans obtained with the Matterport scanners in 6 areas from three different buildings, divided into 271 individual rooms. Each point in the scene is annotated with one label from 13 categories (ceiling, wall, beam, chair, column etc. and clutter), and is represented by its 3D coordinates, RGB features and normalized location. The S3DIS is a highly unbalanced dataset [50], floor, wall, chair and other common furniture items being the dominant classes in the dataset while bookcase, window and beam etc. being the rare classes. To prepare the training

Table 4: Ablation analysis on ScanNet comparing 3, 5 and 7 point-wise spatial attention modules added to our network and comparing the results when 3 point-wise spatial attention modules are added to different stages of our network. We also test adding three attention blocks to standard PointNet++[8].

Block Position	Overall Accuracy (OA %)
LAE-Conv layer (1-7)	84.9
LAE-Conv layer (2,3,4,5,6)	85.7
LAE-Conv layer (3,4,5)	**86.7**
LAE-Conv layer (2,4,6)	86.0
LAE-Conv layer (1,4,7)	85.5
PointNet++[8] (vanilla) baseline	83.3
PointNet++[8] (vanilla, 3-5)	84.7

data, rooms in S3DIS are split into blocks of $2m \times 2m$, with $0.5m$ padding on each direction (x,y). We randomly sample 4096 points from each block during training while all points are used at test time. Similar to PointNet [7], we follow the same 6-fold cross validation strategy across 6 areas. To obtain the overall segmentation accuracy, we evaluate 6 models on their corresponding test areas and report the average results.

For comparison, we use xyz coordinates and RGB information as the point features. Therefore, the input data size to the network is 4096×6. As shown in Figure 3, we use downsampling and upsampling operations from PointNet++ [8] for both encoder and decoder parts. The output point numbers and feature dimensions of different LAE-Conv layers are $(N_1 = 4096, C_1 = 64)$, $(N_2 = 1024, C_2 = 128)$, $(N_3 = 512, C_3 = 256)$, $(N_4 = 128, C_4 = 512)$, $(N_5 = 512, C_5 = 256)$, $(N_6 = 1024, C_6 = 256)$ and $(N_7 = 4096, C_7 = 128)$ respectively. The fully connected layer with size $(N_{fc} = 4096, C_{fc} = 13)$ converts the final features into probability of each class. We set $(m = 1, K = 16)$ during the neighbors search process. For the three point-wise attention modules, the output point numbers and feature dimensions are $(N_{p_1} = 512, C_{p_1} = 256)$, $(N_{p_2} = 128, C_{p_2} = 512)$ and $(N_{p_3} = 512, C_{p_3} = 256)$ respectively. We set the initial learning rate to 0.001, batch size to 32 and momentum to 0.9. We set the decay rate to 0.7 and stop the training process after 1000 epochs.

Table 5 summerizes the quantitative results where our proposed method outperforms the baseline methods PointNet [7], SPGraph [12], RSNet [31], 3DRCNN [51] and PointCNN [11]. It is worth noting that our method achieves higher accuracy for some rare class objects, such as beam, column, window, board and

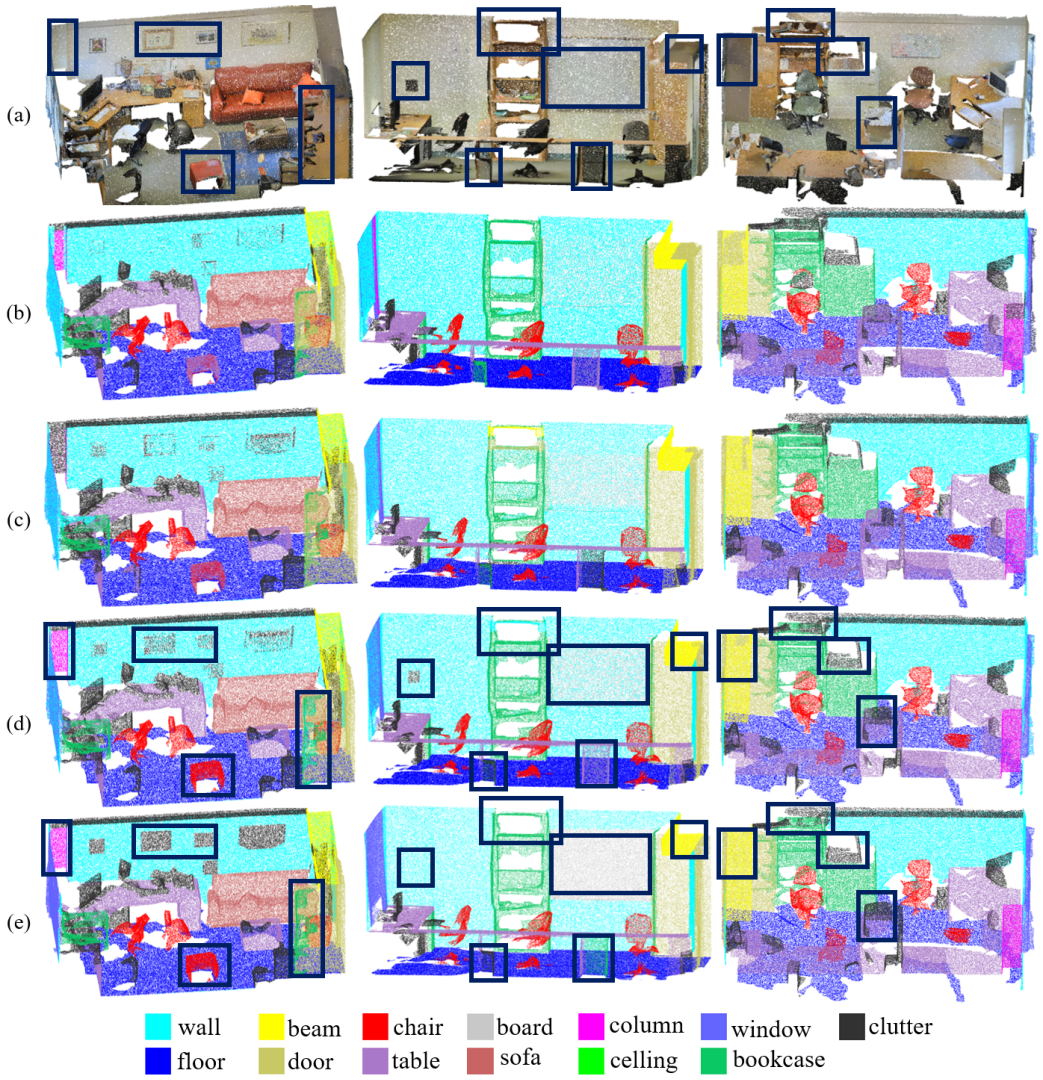

Figure 5: Qualitative comparison on three scenes of S3DIS. (a) Input point cloud with xyz coordinates and RGB features. Semantic segmentation results by (b) PointNet [7], (c) PointCNN [11] and (d) our method. (e) Ground truth Semantic labels. Different colors denote different categories. These scenes contain 13 categories. Boxes highlight some examples where our method performs better than others.

Table 5: Quantitative comparison using overall accuracy (OA,%) and mean IoU (mIoU,%) on S3DIS.

	mIoU	OA	celling	floor	wall	beam	column	window	door	chair	table	bookcase	sofa	board	clutter
PointNet [7]	47.6	78.5	88.0	88.7	69.3	42.4	23.1	47.5	51.6	54.1	42.0	9.6	38.2	29.4	35.2
SPGraph [12]	62.1	85.5	89.9	95.1	76.4	62.8	47.1	55.3	**68.4**	73.5	69.2	**63.2**	45.9	8.70	52.9
RSNet [31]	56.47	-	92.48	92.83	**78.56**	32.75	34.37	51.62	68.11	60.13	59.72	50.22	16.42	44.85	52.03
3DRCNN [51]	53.4	85.7	95.2	**98.6**	77.4	0.80	9.83	52.7	27.9	**78.3**	**76.8**	27.4	58.6	39.1	51.0
PointCNN [11]	65.39	88.14	**94.78**	97.3	75.82	63.25	51.71	58.38	57.18	71.63	69.12	39.08	**61.15**	52.19	58.59
Ours	**66.3**	**88.95**	94.3	97.0	76.02	**64.66**	**53.7**	**59.17**	58.8	72.4	69.2	42.63	60.83	**54.14**	**59.05**

clutter because our method is able to capture more global information of points that are far apart.

In Figure 5, we compare our method with PointNet [7] and PointCNN [11] qualitatively. It is not surprising that chairs are correctly segmented more often by the three baseline methods because their shapes are more consistent, they are small and not easily confused with other objects. We can see that objects such as the whiteboard hung on the wall, column and window embedded in the wall, clutter next to the table and irregular bookcases are quite difficult to segment. We also use boxes to mark some examples where our method outperformed the baseline methods. In the first scene, our method obtains more regular segmentation of the painting on the wall than PointNet [7] and PointCNN [11]. Our final segmentation result preserves the full shapes of column and bookcase next to wall while other methods mistake them for wall or clutter. We also obtain a smoother prediction for chair in the front row than the other methods. In the second scene, the board on the wall is more accurately estimated by our method compared to PointNet and PointCNN. Our method makes fewer mistakes in predicting the bookshelves, beam and clutter which are up and below the table compared to other approaches. Notably, our method can predict the clutter on the left wall, even though it is not marked by ground truth. In the third scene, our method also outputs fewer incorrect predictions for bookcase, table, chair and beam compared to other approaches.

5.3. ShapeNet

We also extend our network architecture to perform part segmentation on the ShapeNet dataset [52], which consists of 16881 shape models from 16 object categories. Each object in ShapeNet is annotated with 2 to 5 parts. We follow the settings from [48] to divide the ShapeNet dataset for training, validation and testing. During training, we randomly sample 2048 points from each 3D shape while all points from each 3D shape are used during the test stage.

Table 6: Quantitative comparison on ShapeNet part dataset [52]. The values show part-averaged IoU (pIoU%), mean per-category pIoU (mpIoU%) and per-category IoU (%) scores.

	pIoU	mpIoU	air plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate board	table
shapes			2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152	5271
PointNet [7]	83.7	80.4	83.4	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
PointNet++ [8]	85.1	81.9	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
DGCNN [9]	85.1	82.3	84.2	83.7	84.4	77.1	90.9	78.5	91.5	87.3	82.9	96.0	67.0	93.3	82.6	59.7	75.5	82.0
RSNet [31]	84.9	81.4	82.7	86.4	84.1	78.2	90.4	69.3	91.4	87.0	83.5	95.4	66.0	92.6	81.8	56.1	75.8	82.2
SGPN [40]	85.8	82.8	80.4	78.6	78.8	71.5	88.6	78.0	90.9	83.0	78.8	95.8	77.8	93.8	**87.4**	60.1	**92.3**	**89.4**
ASCNet [53]	84.6	81.78	83.8	80.8	83.5	79.3	90.5	69.8	91.7	86.5	82.9	96.0	69.2	93.8	82.5	62.9	74.4	80.8
PCNNet [30]	85.1	81.8	82.4	80.1	85.5	79.5	90.8	73.2	91.3	86.0	85.0	95.7	73.2	94.8	83.3	51.0	75.0	81.8
PGrid [13]	**86.4**	82.23	85.7	82.5	81.8	77.9	92.1	82.4	**92.7**	85.8	84.2	95.3	65.2	93.4	81.7	56.9	73.5	84.6
SPLATNet [29]	85.4	83.69	83.2	84.3	**89.1**	80.3	90.7	75.5	92.1	87.1	83.9	**96.3**	75.6	95.8	83.8	64.0	75.5	81.8
KCGP [39]	84.7	82.21	82.8	81.5	86.4	77.6	90.3	76.8	91.0	87.2	84.5	95.5	69.2	94.4	81.6	60.1	75.2	81.3
SpiderCNN [14]	85.3	81.7	83.5	81	87.2	77.5	90.7	76.8	91.1	87.3	83.3	95.8	70.2	93.5	82.7	59.7	75.8	82.8
SONet [54]	84.9	81.0	82.8	77.8	88.0	77.3	90.6	73.5	90.7	83.9	82.8	94.8	69.1	94.2	80.9	53.1	72.9	83.0
SSCN [23]	85.98	83.3	**84.1**	83.0	84.0	**80.8**	**91.4**	78.2	91.6	**89.1**	85.0	95.8	73.7	95.2	84.0	58.5	76.0	82.7
PointCNN [11]	86.1	**84.6**	**84.1**	**86.5**	86.0	**80.8**	90.6	79.7	92.3	88.4	85.3	96.1	77.2	**95.3**	84.2	64.2	80.0	83.0
Ours	85.9	84.10	83.3	86.1	85.7	80.3	90.5	**82.7**	91.5	88.1	**85.5**	95.9	**77.9**	95.1	84.0	**64.3**	77.6	82.8

For a fair comparison, we only use the xyz coordinates as the point features. The size of input data for the network is 2048×3. The network architecture is illustrated in Figure 3, we adjust the network parameters to suit ShapeNet. The output point numbers and feature dimensions of different LAE-Conv layers are $(N_1 = 2048, C_1 = 64)$, $(N_2 = 1024, C_2 = 128)$, $(N_3 = 256, C_3 = 256)$, $(N_4 = 128, C_4 = 512)$, $(N_5 = 256, C_5 = 256)$, $(N_6 = 1024, C_6 = 256)$ and $(N_7 = 2048, C_7 = 128)$ respectively. A fully connected layer with size $(N_{fc} = 128, C_{fc} = 16)$ is used at the end to convert the point features into part predictions. We set $(m = 1, K = 16)$ for the neighborhood search. For the three point-wise attention block, the output point numbers and feature dimensions are $(N_{p_1} = 256, C_{p_1} = 256)$, $(N_{p_2} = 128, C_{p_2} = 512)$ and $(N_{p3} = 256, C_{p_3} = 256)$ respectively. We set the initial learning rate to 0.003, batch size to 16, momentum to 0.9, decay rate to 0.7 and stop the training after 500 epochs.

We use the same evaluation metric (mean IoU) on points as PointNet [7] to compare our method with others methods [7, 8, 9, 31, 40, 53, 30, 13, 29, 39, 14, 54, 23, 11]. We report the part-averaged IoU (pIoU%), mean per-category pIoU (mpIoU%) and per-category IoU (%) scores in Table 6. Our method achieves on par performance with most methods in the metrics pIoU and mpIoU. In individual categories, we rank the best in ear phone, lamp, motor and rocket. As we can see, our method performs better when there are fewer data points as in the case of ear phone, motor and rocket.

6. Conclusion

We proposed a point attention network for 3D point cloud semantic segmentation. Our network adaptively integrates local point features and long-range contextual information. We introduced a novel local attention-edge convolution (LAE-Conv) layer which exploits attention mechanism on a local graph constructed by the central point and its neighborhood to capture accurate and robust geometric details. To refine the output local features of LAE-Conv layer, we proposed a point-wise spatial attention module and showed that this module can generalize to other networks to improve their accuracy. Finally, we adapted the U-shaped network to combine the LAE-Conv layer and point-wise spatial attention modules. Experiments on challenging benchmark datasets show that our method quantitatively and qualitatively obtains on pair or better performance than existing state-of-the-art in 3D point cloud semantic segmentation.

Acknowledgment

This work was supported in part by National Natural Science Foundation of China under Grant 61573134, Grant 61973106 and in part by the Australian Research Council (ARC) grant DP190102443. Thank Yifeng Zhang and Tingting Yang from Hunan University for helping with baseline experiments setup.

Reference

References

[1] M. Feng, Y. Wang, J. Liu, L. Zhang, H. F. Zaki, A. Mian, Benchmark data set and method for depth estimation from light field images, IEEE Transactions on Image Processing 27 (2018) 3586–3598.

[2] Y. Zou, X. Wang, T. Zhang, B. Liang, J. Song, H. Liu, Broph: An efficient and compact binary descriptor for 3d point clouds, Pattern Recognition 76 (2018) 522–536.

[3] Y. Lei, Z. Zhou, P. Zhang, Y. Guo, Z. Ma, L. Liu, Deep point-to-subspace metric learning for sketch-based 3d shape retrieval, Pattern Recognition 96 (2019) 106981.

[4] H. Liu, Y. Cong, C. Yang, Y. Tang, Efficient 3d object recognition via geometric information preservation, Pattern Recognition 92 (2019) 135–145.

[5] A. Nurunnabi, G. West, D. Belton, Outlier detection and robust normal-curvature estimation in mobile laser scanning 3d point cloud data, Pattern Recognition 48 (2015) 1404–1419.

[6] Z. Wang, F. Lu, Voxsegnet: Volumetric cnns for semantic part segmentation of 3d shapes, IEEE transactions on visualization and computer graphics (2019).

[7] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 652–660.

[8] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, Advances in Neural Information Processing Systems (NeurIPS) (2017) 5099–5108.

[9] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, Dynamic graph cnn for learning on point clouds, ACM Transactions on Graphics (TOG) (2019).

[10] M. Simonovsky, N. Komodakis, Dynamic edge-conditioned filters in convolutional neural networks on graphs, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 3693–3702.

[11] Y. Li, R. Bu, M. Sun, B. Chen, Pointcnn, Advances in Neural Information Processing Systems (NeurIPS) (2018) 820–830.

[12] L. Landrieu, M. Simonovsky, Large-scale point cloud semantic segmentation with superpoint graphs, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 4558–4567.

[13] T. Le, Y. Duan, Pointgrid: A deep network for 3d shape understanding, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 9204–9214.

[14] Y. Xu, T. Fan, M. Xu, L. Zeng, Y. Qiao, Spidercnn: Deep learning on point sets with parameterized convolutional filters, Proceedings of the European Conference on Computer Vision (ECCV) (2018) 87–102.

[15] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3d shapenets: A deep representation for volumetric shapes, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015) 1912–1920.

[16] B. Graham, M. Engelcke, L. van der Maaten, 3d semantic segmentation with submanifold sparse convolutional networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 18–22.

[17] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, L. J. Guibas, Volumetric and multi-view cnns for object classification on 3d data, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 5648–5656.

[18] Y. Zhou, O. Tuzel, Voxelnet: End-to-end learning for point cloud based 3d object detection, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 4490–4499.

[19] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, X. Tong, O-cnn: Octree-based convolutional neural networks for 3d shape analysis, ACM Transactions on Graphics (TOG) 36 (2017) 72.

[20] G. Riegler, A. O. Ulusoy, A. Geiger, Octnet: Learning deep 3d representations at high resolutions, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 3577–3586.

[21] R. Klokov, V. Lempitsky, Escape from cells: Deep kd-networks for the recognition of 3d point cloud models, Proceedings of the IEEE Conference on Computer Vision (ICCV) (2017) 863–872.

[22] H. Lei, N. Akhtar, A. Mian, Octree guided cnn with spherical kernels for 3d point clouds, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 9631–9640.

[23] B. Graham, M. Engelcke, L. van der Maaten, 3d semantic segmentation with submanifold sparse convolutional networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 18–22.

[24] M. Ren, A. Pokrovsky, B. Yang, R. Urtasun, Sbnet: Sparse blocks network for fast inference, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 8711–8720.

[25] R. Roveri, L. Rahmann, C. Oztireli, M. Gross, A network architecture for point cloud classification via automatic depth images generation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 4176–4184.

[26] A. Dai, M. Niessner, 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation, Proceedings of the European Conference on Computer Vision (ECCV) (2018) 452–468.

[27] H. You, Y. Feng, R. Ji, Y. Gao, Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition, ACM Multimedia Conference on Multimedia Conference (2018) 1310–1318.

[28] H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller, Multi-view convolutional neural networks for 3d shape recognition, Proceedings of the IEEE Conference on Computer Vision (ICCV) (2015) 945–953.

[29] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, J. Kautz, Splatnet: Sparse lattice networks for point cloud processing, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 2530–2539.

[30] M. Atzmon, H. Maron, Y. Lipman, Point convolutional neural networks by extension operators, ACM Transactions on Graph 37 (2018).

[31] Q. Huang, W. Wang, U. Neumann, Recurrent slice networks for 3d segmentation of point clouds, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 2626–2635.

[32] D. Rethage, J. Wald, J. Sturm, N. Navab, F. Tombari, Fully-convolutional point networks for large-scale point clouds, Proceedings of the European Conference on Computer Vision (ECCV) (2018) 596–611.

[33] M. Tatarchenko, J. Park, V. Koltun, Q.-Y. Zhou, Tangent convolutions for dense prediction in 3d, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 3887–3896.

[34] J. Bruna, W. Zaremba, A. Szlam, Y. Lecun, Spectral networks and locally connected networks on graphs, International Conference on Learning Representations (ICLR) (2014) 688.

[35] C. Wang, B. Samari, K. Siddiqi, Local spectral graph convolution for point set feature learning, Proceedings of the European Conference on Computer Vision (ECCV) (2018) 52–66.

[36] G. Te, W. Hu, Z. Guo, A. Zheng, Rgcnn: Regularized graph cnn for point cloud segmentation, ACM Multimedia Conference on Multimedia Conference (2018) 746–754.

[37] M. Jaderberg, K. Simonyan, A. Zisserman, et al., Spatial transformer networks, Advances in neural information processing systems (NeurIPS) (2015) 2017–2025.

[38] F. Engelmann, T. Kontogianni, A. Hermans, B. Leibe, Exploring spatial context for 3d semantic segmentation of point clouds, Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017) 716–724.

[39] Y. Shen, C. Feng, Y. Yang, D. Tian, Mining point cloud local structures by kernel correlation and graph pooling, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 4548–4557.

[40] W. Wang, R. Yu, Q. Huang, U. Neumann, Sgpn: Similarity group proposal network for 3d point cloud instance segmentation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 2569–2578.

[41] B.-S. Hua, M.-K. Tran, S.-K. Yeung, Pointwise convolutional neural networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 984–993.

[42] H. Zhao, L. Jiang, C. Fu, J. Jia, Pointweb: Enhancing local neighborhood features for point cloud processing, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 5565–5573.

[43] W. Wu, Z. Qi, L. Fuxin, Pointconv: Deep convolutional networks on 3d point clouds, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 9621–9630.

[44] L. Wang, Y. Huang, Y. Hou, S. Zhang, J. Shan, Graph attention convolution for point cloud semantic segmentation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 10296–10305.

[45] P. Velikovi, G. Cucurull, A. Casanova, A. Romero, P. Li, Y. Bengio, Graph attention networks, International Conference on Learning Representations (ICLR) (2018).

[46] X. Wang, R. Girshick, A. Gupta, K. He, Non-local neural networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 7794–7803.

[47] J. Fu, J. Liu, H. Tian, H. Lu, Dual attention network for scene segmentation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 3146–3154.

[48] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al., Shapenet: An information-rich 3d model repository, arXiv preprint arXiv:1512.03012 (2015).

[49] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Niessner, Scannet: Richly-annotated 3d reconstructions of indoor scenes, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 5828–5839.

[50] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3d semantic parsing of large-scale indoor spaces, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 1534–1543.

[51] X. Ye, J. Li, H. Huang, L. Du, X. Zhang, 3d recurrent neural networks with context fusion for point cloud semantic segmentation, Proceedings of the European Conference on Computer Vision (ECCV) (2018) 403–417.

[52] L. Yi, V. G. Kim, D. Ceylan, I. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas, et al., A scalable active framework for region annotation in 3d shape collections, ACM Transactions on Graphics (TOG) 35 (2016) 210.

[53] S. Xie, S. Liu, Z. Chen, Z. Tu, Attentional shapecontextnet for point cloud recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 4606–4615.

[54] J. Li, B. M. Chen, G. H. Lee, So-net: Self-organizing network for point cloud analysis, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 9397–9406.