

Graph Attention Convolution for Point Cloud Semantic Segmentation

Lei Wang¹, Yuchun Huang^{1*}, Yaolin Hou¹, Shenman Zhang¹, Jie Shan^{2*}
¹Wuhan University, China ²Purdue University, USA

{wlei, hycwhu, houyaolin, smzhang}@whu.edu.cn, jshan@purdue.edu

Abstract

Standard convolution is inherently limited for semantic segmentation of point cloud due to its isotropy about features. It neglects the structure of an object, results in poor object delineation and small spurious regions in the segmentation result. This paper proposes a novel graph attention convolution (GAC), whose kernels can be dynamically carved into specific shapes to adapt to the structure of an object. Specifically, by assigning proper attentional weights to different neighboring points, GAC is designed to selectively focus on the most relevant part of them according to their dynamically learned features. The shape of the convolution kernel is then determined by the learned distribution of the attentional weights. Though simple, GAC can capture the structured features of point clouds for fine-grained segmentation and avoid feature contamination between objects. Theoretically, we provided a thorough analysis on the expressive capabilities of GAC to show how it can learn about the features of point clouds. Empirically, we evaluated the proposed GAC on challenging indoor and outdoor datasets and achieved the state-of-the-art results in both scenarios.

1. Introduction

Semantic segmentation of point clouds aims to assign a category label to each point, which is an important yet challenging task for 3D understanding. Recent approaches have attempted to generalize convolutional neural network (CNN) from grid domains (i.e., speech signals, images, and video data) to unorganized point clouds [34, 45, 33, 35, 44, 36, 26, 14]. However, due to the isotropy of their convolution kernels about the neighboring points' feature attributes, these works are inherently limited for the semantic point cloud segmentation. Intuitively, the learned features for the points at the boundary of two objects (i.e., point 1 in Figure 1) are actually from both objects rather than the object they truly belong to, which results in ambiguous label as-

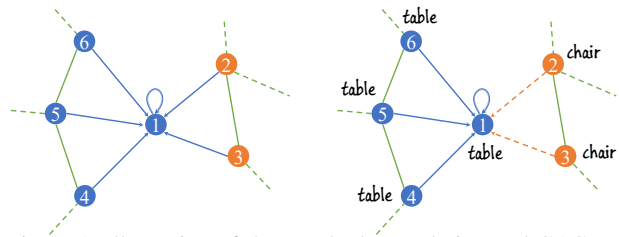


Figure 1. Illustration of the standard convolution and GAC on a subgraph of a point cloud. **Left:** The weights of standard convolution are determined by the neighbors' spatial positions, and the learned feature at point 1 characterizes all of its neighbors indistinguishably. **Right:** In GAC, the attentional weights on "chair" (the brown dotted arrows) are masked, so that the convolution kernel can focus on the "table" points.

segment.

In fact, standard convolution kernels work in a regular receptive field for feature response, and the convolution weights are fixed at specific positions within the convolution window. This kind of position-determined weights results in the isotropy of the convolution kernel about the feature attributes of neighboring points. For instance, in Figure 1 the learned feature at point 1 characterizes its neighboring "table" and "chair" indistinguishably. This limitation of the standard convolution neglects the structural connection between points belonging to the same object, and results in poor object delineation and small spurious regions in the segmentation result.

To address this problem, the key idea of this work is as follows. Based on the position-determined weights of the standard convolution, we learn to mask or weaken part of the convolution weights according to the neighbors' feature attributes, so that the actual receptive field of our convolution kernel for point clouds is no longer a regular 3D box but has its own shape to dynamically adapt to the structure of the objects.

In this paper, we implement this idea by proposing a novel GAC to selectively focus on the most relevant part of the neighbors in the receptive field. Specifically, inspired by the idea of attention mechanism [4, 13, 47], GAC is designed to dynamically assign proper attentional weights to different neighboring points by combining their spatial po-

*Corresponding author.

sitions and feature attributes. The shape of the convolution kernel is then determined by the learned distribution of the attentional weights.

Finally, like the standard convolution in grid domain, our GAC can also be efficiently implemented on the graph representation of a point cloud. Referring to image segmentation network, we train an end-to-end graph attention convolution network (GACNet) with the proposed GAC for semantic point cloud segmentation.

Notably, postprocessing of CNN’s outputs using conditional random field (CRF) has practically become a *de facto* standard in semantic segmentation [45, 5, 9, 2]. However, by combining the spatial and feature constraints for attentional weights generation, GAC shares the same properties as CRF, which encourages the label agreement between similar points. Thus, CRF is no longer needed in the proposed GACNet.

Our contributions are as follows:

- We propose a novel graph attention convolution with learnable kernel shapes to dynamically adapt to the structure of the objects;
- We provide thorough theoretical and empirical analyses on the capability and effectiveness of the proposed graph attention convolution;
- We train an end-to-end graph attention convolution network for point cloud semantic segmentation with the proposed GAC and experimentally demonstrate its effectiveness.

2. Related Works

This section will discuss the related prior works in three main aspects: deep learning on point clouds, convolution on graphs, and CRF in deep learning.

Deep learning on point clouds. While deep learning has been successfully used in 2D images, there are still many challenges to exploring its feature learning power for 3D point clouds with irregular data structures. Recent researches on this issue can be mainly summarized as voxelization-based [25, 49], multi-view-based [43, 24], graph-based [7, 51, 42] and set-based methods [33, 35].

The voxelization-based method [50, 30] aims to discretize the point cloud space into regular volumetric occupancy grids, so that the 3D convolution can be applied similarly as the image. These full-voxel-based methods inevitably lead to information loss, as well as memory and computational consumption as it increases cubically with respect to the voxel’s resolution. To reduce the computational cost of these full-voxel-based methods, OctNet [38] and Kd-Net [20] were designed to resolve them by skipping the computations on empty voxels and focusing on informative voxels. The multi-view-based method [43, 24, 18] represents the point cloud as a set of images rendered from

multiple views. However, it is still unclear how to determine the number and distribution of the views to cover the 3D objects while avoiding mutual occlusions.

The graph-based method [7, 51] first represents the point cloud as a graph according to their spatial neighbors, and then generalizes the standard CNN to adapt to the graph-structural data. Shen et al. [40] defined a point-set kernel as a set of learnable 3D points that jointly respond to the neighboring points according to their geometric affinities measured by the kernel correlation. 3DGNN [36] applied graph neural network to RGBD data. However, due to the isotropy of its aggregation function, 3DGNN can hardly adapt to objects with different structures. ECC [42] and SPG [23] proposed to generate the convolution filters according to the edge labels (weights), so that the information can propagate in a specific direction on the graph. Nevertheless, ECC and SPG can only capture some specific structures since these edge labels (weights) are predefined.

Benefiting from the development of deep learning on sets [33, 52, 37], researchers recently constructed effective and simple architecture to directly learn on point sets by first computing individual point features from per-point multi-layer perceptron (MLP) and then aggregating all the features as a global presentation of a point cloud [35, 12]. The set-based method can be used directly on the point level and is robust to the rigid transformation. However, it neglects the spatial neighboring relation between points, which contains fine-grained structural information for semantic segmentation.

Convolution on Graphs. Related works about convolution on graphs can be categorized as spectral approaches and non-spectral approaches. Spectral approaches work with a spectral representation of graphs that relies on the eigen-decomposition of their Laplacian matrix [19, 10]. The corresponding eigenvectors can be regarded as the Fourier bases in the harmonic analysis of spectral graph theory. The spectral convolution can then be defined as the element-wise product of two signals’ Fourier transform on the graph [8]. This spectral convolution does not guarantee the spatial localization of the filter and thus requires expensive computations [41, 17]. In addition, as spectral approaches are associated with their corresponding Laplacian matrix, a spectral CNN model learned on one graph cannot be transferred to another graph that has a different Laplacian matrix.

Non-spectral approaches aim to define convolution directly on a graph with local neighbors in a spatial or manifold domain. The key to non-spectral approaches is to define a set of sharing weights applied to the neighbors of each vertex [3, 48]. Duvenaud et al. [11] computed a weight matrix for each vertex and multiplied it to its neighbors following a sum operation. Niepert et al. [32] proposed selecting and ordering the neighbors of each vertex heuristically

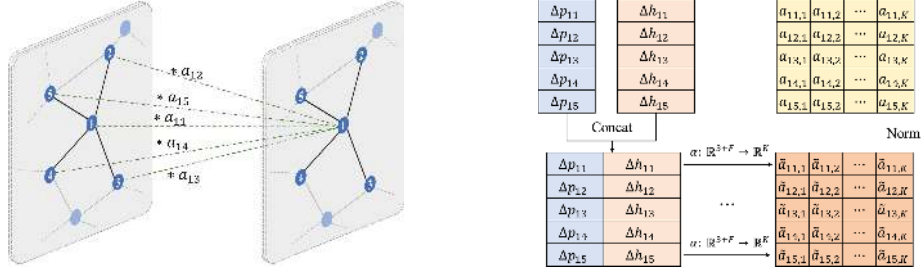


Figure 2. **Left:** Illustration of GAC on a subgraph of a point cloud. The output is a weighted combination of the neighbors of point 1. **Right:** The attention mechanism employed in GAC for dynamically attentional weights generating. It receives the neighboring vertices’ spatial positions and features as input, and then maps them to normalized attentional weights.

so that the 1D CNN can be used. Monti et al. [31] proposed a unified framework that allows the generalization of CNN architecture to graph using fixed local polar pseudo-coordinates around each vertex. Hamilton et al. [16] introduced an inductive framework by applying a specific aggregator over the neighbors, such as the max/mean operator or a recurrent neural network (RNN). However, their convolution weights are mainly generated according to the predefined local coordinate system, while neglecting the structure of the objects for semantic segmentation.

CRF in Deep Learning. CRF [22] possesses fine-grained probabilistic modeling capability, while CNN has powerful feature learning capability. The combination of CRF and CNN has been proposed in many image segmentation works [5, 9, 2, 29]. Recently, referring to the mean-field algorithm [21], the iteration of CRF inference was modeled as a stack of CNN layers [53, 28]. For 3D point cloud, following CRF-RNN [53], SegCloud [45] extends the implementation of CRF into 3D point clouds after a fully connected CNN. However, since CRF is applied as an individual part following the CNN, it is difficult to explore the power of their combination.

3. Method

We propose a novel graph attention convolution (GAC) for structured feature learning of 3D point cloud and demonstrate its theoretical advantage (Section 3.1). Afterwards, we construct an end-to-end point cloud segmentation framework (Section 3.2) with the proposed GAC. The details of converting point cloud into our needed graph pyramid are provided in Section 3.3.

3.1. Graph attention convolution

Consider a graph $G(V, E)$ constructed from a given point cloud $P = \{p_1, p_2, \dots, p_N\} \in \mathbb{R}^3$ according to their spatial neighbors, where $V = \{1, 2, \dots, N\}$ and $E \subseteq |V| \times |V|$ represent the set of vertices and edges respectively and N is the number of vertices (points). Denote $\mathcal{N}(i) = \{j : (i, j) \in E\} \cup \{i\}$ (including itself) as the neighbor set of vertex i . Let $H = \{h_1, h_2, \dots, h_N\}$ be a set

of input vertex features, each feature $h_i \in \mathbb{R}^F$ is associated with a corresponding graph vertex $i \in V$, where F is the feature dimension of each vertex.

Our GAC is designed to learn a function $g: \mathbb{R}^F \rightarrow \mathbb{R}^K$, which maps the input features H to a new set of vertex features $H' = \{h'_1, h'_2, \dots, h'_N\}$ with $h'_i \in \mathbb{R}^K$, while maintaining the structural connection between these output features. Meanwhile, unlike the relatively fixed neighboring relation in image domain, the proposed GAC should also be able to handle the unordered and size-varying neighbors while retaining the weight sharing property.

To this end, we construct a sharing attention mechanism $\alpha: \mathbb{R}^{3+F} \rightarrow \mathbb{R}^K$ to focus on the most relevant part of the neighbors for feature learning, so that the convolution kernel of GAC can dynamically adapt to the structure of the objects. Specifically, the attentional weight of each neighboring vertex is computed as follows:

$$\tilde{a}_{ij} = \alpha(\Delta p_{ij}, \Delta h_{ij}), j \in \mathcal{N}(i) \quad (1)$$

where $\tilde{a}_{ij} = [\tilde{a}_{ij,1}, \tilde{a}_{ij,2}, \dots, \tilde{a}_{ij,K}] \in \mathbb{R}^K$ indicates the attentional weight vector of vertex j to vertex i . $\Delta p_{ij} = p_j - p_i$, and $\Delta h_{ij} = M_g(h_j) - M_g(h_i)$, where $M_g: \mathbb{R}^F \rightarrow \mathbb{R}^K$ is a feature mapping function applied on each vertex, i.e., M_g is a multilayer perceptron. The first term of α indicates the spatial relations of the neighboring vertices, which helps to span the unordered neighbors to meaningful surface. The second term measures the feature difference between vertex pairs, which guides us to assign more attention to the similar neighbors. The sharing attention mechanism α can be implemented with any differentiable architecture, we use a multilayer perceptron in this work (as shown in Figure 2), which can be formulated as follows:

$$\alpha(\Delta p_{ij}, \Delta h_{ij}) = M_\alpha([\Delta p_{ij} || \Delta h_{ij}]) \quad (2)$$

where $||$ is the concatenation operation, M_α indicates the applied multilayer perceptron.

In addition, to handle the size-varying neighbors across different vertices and spatial scales, the attentional weights are normalized across all the neighbors of vertex i as fol-

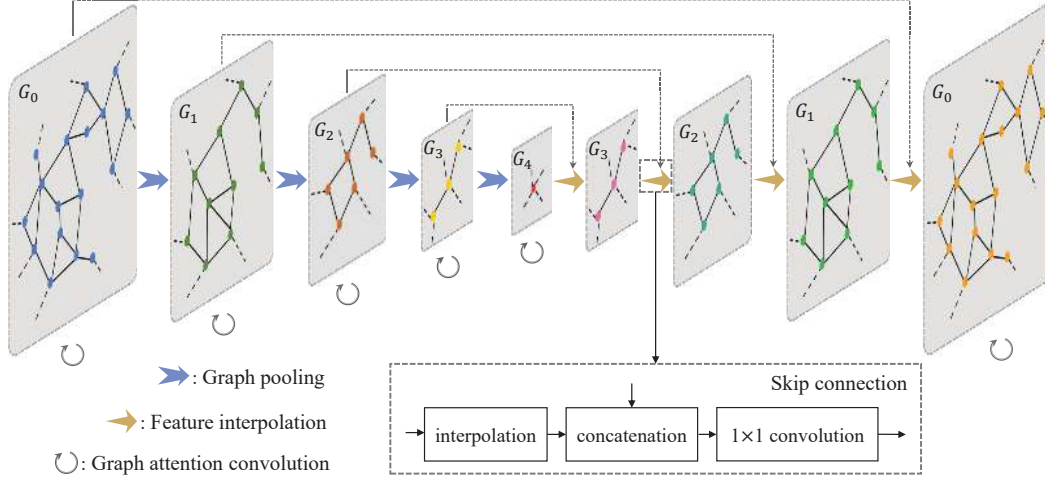


Figure 3. GACNet architecture. Our GACNet is constructed on the graph pyramid of a point cloud. On each scale of the graph pyramid, the proposed GAC is applied for local feature learning, followed by the graph pooling for resolution reducing in each feature channel. After that, the learned features are interpolated back to the finest scale layer by layer for point-wise label assignment.

lows:

$$a_{ij,k} = \frac{\exp(\tilde{a}_{ij,k})}{\sum_{l \in \mathcal{N}(i)} \exp(\tilde{a}_{il,k})} \quad (3)$$

where $\tilde{a}_{ij,k}$ is the attentional weight of vertex j to vertex i at the k -th feature channel.

Therefore, the final output of the proposed GAC can be formulated as follows:

$$h'_i = \sum_{j \in \mathcal{N}(i)} a_{ij} * M_g(h_j) + b_i \quad (4)$$

where $*$ represents the Hadamard product, which produces the element-wise production of two vectors, and $b_i \in \mathbb{R}^K$ is a learnable bias.

Relationship to standard convolution. The convolution weights of a standard convolution in the grid domain are determined by the neighbors' local spatial positions. In our GAC, the attentional weights are generated according to not only the neighbors' spatial positions but also their dynamically learned features. Additionally, as GAC is designed on the spatial neighbors of points, it also retains the key properties of the standard convolution in grid domain: weight sharing and locality.

Relationship to prior works. The proposed GAC is related to several prior works, mainly including GAT [47] and PointNet [33].

Although we are inspired by the idea of attention mechanism as GAT [47], our GAC is different: 1) GAC assigns proper attentional weights to not only different neighboring points but also different feature channels, as the features at different channels are hopefully independent; 2) Compared to GAT, GAC incorporates the local spatial relationship between neighboring points, which plays an important role in 3D shape analysis; 3) We generate the attentional weights

based on the feature differences rather than the concatenation of two neighboring features, which is more efficient and explicit to characterize the feature relation.

PointNet [33] and its variations [35] have achieved promising results for point cloud analysis by directly learning on point sets. Key to PointNet is the use of a max-pooling operator (including a MLP). It can be seen as an extreme case of GAC as “max attention”, which aggregates the neighboring features by taking the max value at each feature channel. The max operator tends to capture the most “special” features, which damages the structural connections between the points of an object and becomes sensitive to noise. Comparatively, the proposed GAC aggregates the neighboring features by assigning them proper attentional weights, maintaining the structure of the objects for fine-grained point cloud segmentation.

Theoretical analysis. In this section, we explore the expressive capabilities of our GAC to further understand how GAC can efficiently learn the features of point clouds. Specifically, we consider whether GAC can learn to precisely represent the neighboring features of each vertex.

Suppose the input vertex features H are bounded, i.e., $H \subseteq [a, b]^F$, where a and b indicate the lower and upper bound respectively. In fact, we can show that the proposed GAC is capable of aggregating the entire neighbor information of any vertex on the graph $G(V, E)$ to an arbitrary precision:

Theorem 1. *Let $\mathcal{X} = \{S : S \subseteq [a, b]^F \text{ and } S \text{ is finite}\}$, $f : \mathcal{X} \rightarrow \mathbb{R}$ is a continuous set function w.r.t. Hausdorff distance $d_H(\cdot, \cdot)$. Denote $S_i = \{h_j : j \in \mathcal{N}(i) \in \mathcal{X}\}$ as the set of neighboring points of vertex $i \in V$ with arbitrary order. $\forall \epsilon > 0, \exists K \in \mathbb{Z}$ and parameter θ of GAC, such that*

for any $i \in V$,

$$|f(S_i) - \gamma(g_\theta(S_i))| < \epsilon \quad (5)$$

where γ is a continuous function, and $g_\theta(S_i) \in \mathbb{R}^K$ is the output of GAC.

The full proof is provided in the Appendix. Similar to PointNet, in the worst case, our GAC can learn to divide the point cloud into a volumetric representation. In PointNet, the representation capability is limited by the output dimension K . However, as the attention mechanism in our GAC actually acts as a feature encoder, GAC is capable of approximating the set function f even when K is not sufficiently large.

3.2. Graph attention convolution network

We follow the common image segmentation architecture to organize our network for point cloud semantic segmentation, coined graph attention convolution network (GACNet). The difference is that, our GACNet is implemented on the graph pyramid of a point cloud, as shown in Figure 3. At each scale of the graph pyramid, GAC is applied for local feature learning. Then a graph pooling operation is used to reduce the resolution of point clouds in each feature channel. After that, the learned features are interpolated back to the finest scale layer by layer. Inspired by [27], features at the same scale are skip-connected. Finally, considering the loss of feature fidelity caused by the multiple graph pooling and feature interpolation layers, an additional GAC layer is applied at the finest scale for feature refinement.

Graph pooling. Graph pooling aims to output the aggregated features on the vertices of a coarsened graph. Denote H'_l as the output feature set at the l -th scale of the graph pyramid, the input feature set H_{l+1} of the $(l+1)$ -th scale is calculated as follows:

$$h_v = \text{pooling}\{h'_j : j \in \mathcal{N}_l(v)\} \quad (6)$$

where $h_v \in H_{l+1}$ and $\mathcal{N}_l(v)$ indicates the neighbors of vertex v at the l -th scale. The *pooling* function can be a max or mean function, which corresponds to the max and mean pooling, respectively [42].

Feature interpolation. To finally obtain the feature map that has the same number of points as the original input, we must interpolate the learned features from the coarsest scale to the original scale layer by layer. Let H'_l be the learned feature set at the l -th scale of the graph pyramid, P_l and P_{l-1} are the spatial coordinates set of the l -th and $(l-1)$ -th scales, respectively. To obtain the features of the $(l-1)$ -th scale, we simply search the three nearest neighbors of P_{l-1} in P_l and calculate the weighted sum of their features. The combination weights are calculated according to the neighbors' normalized spatial distances [35].

GACNet vs. CRF. CRF has practically become a *de facto* standard as the postprocessing of the CNN's outputs

in semantic segmentation tasks. The key idea of CRF is to encourage similar points to share consistent labels. Intuitively, spatially close and appearance-similar points are encouraged to be assigned the same label.

In fact, our GAC shares the same characteristics as the CRF model. Specifically, GAC assigns neighbors proper attentional weights according to both their spatial positions and feature attributes. The spatial position term encourages the spatially close points to share similar features, whereas the feature attribute term aims at leading the information propagating between points with similar attributes (i.e., low-level local features or high-level semantic labels). Therefore, the CRF model is no longer needed in GACNet.

Notably, compared to formulating the CRF model as a recurrent network [53], the proposed GACNet has several compelling advantages. First, rather than using CRF for a postprocessing which is independent of the CNN, GACNet is equivalent to unfolding the recurrent network of CRF into each layer of the network, which directly guides the learned features to maintain the structure of an object for semantic segmentation. Second, compared to the simple message passing and compatibility transform in the class-probability space of CRF [21, 53], GAC also has the capability to map the input signals into a hidden feature space for further feature extraction. We experimentally evaluate these claims in Section 4.3.

3.3. Graph pyramid construction on a point cloud

This section describes how we construct the graph pyramid on point clouds. Specifically, we search the spatial neighbors for all points and link them as a graph. The graph pyramid with different spatial scales is constructed by alternately applying graph construction and coarsening techniques. Additionally, the covariance matrix of each point's neighbors at the finest scale are recorded during the graph construction process, and its eigenvalues are used as local geometrical feature (geo-feature). The initial feature vector of a point is composed of height, RGB, and geo-feature.

Graph construction on a point cloud. For given point cloud P , which records the spatial coordinates of the points, we construct a directed graph $G(V, E)$. Here, each vertex is associated with a point, and the edges are added between the point and its K_G neighbors. In our experiments, the K_G neighbors are randomly sampled within radius ρ , which shows better performance than searching their K_G nearest neighbors as it is unrelated to the density of the point cloud.

Graph coarsening. Similar to pyramid construction in the image domain, we subsample the input point cloud P with a set of ratios using the furthest point sampling algorithm [35]. Denote the subsampled point clouds as $\mathcal{P} = \{P_0, P_1, \dots, P_L\}$, where L is the number of scales for subsampling and $P_0 = P$. For each $P_l (l = 0, \dots, L)$, a corresponding graph $G_l(V_l, E_l)$ can be constructed as de-

Method	OA	mIoU	ceiling	floor	wall	beam	column	window	door	chair	table	bookcase	sofa	board	clutter
PointNet [33]	-	41.09	88.80	97.33	69.80	0.05	3.92	46.26	10.76	52.61	58.93	40.28	5.85	26.38	33.22
SegCloud [45]	-	48.92	90.06	96.05	69.86	0.00	18.37	38.35	23.12	75.89	70.40	58.42	40.88	12.96	41.60
SPG [23]	86.38	58.04	89.35	96.87	78.12	0.00	42.81	48.93	61.58	84.66	75.41	69.84	52.60	2.10	52.22
GACNet(ours)	87.79	62.85	92.28	98.27	81.90	0.00	20.35	59.07	40.85	78.54	85.80	61.70	70.75	74.66	52.82

Table 1. Results on the S3DIS dataset (testing on Area 5 and training on the rest five areas).

Method	OA	mIoU	man-made terrain	natural terrain	high vegetation	low vegetation	buildings	hard scape	scanning artefacts	cars
SnapNet [6]	88.6	59.1	82.0	77.3	79.7	22.9	91.1	18.4	37.3	64.4
SegCloud [45]	88.1	61.3	83.9	66.0	86.0	40.5	91.1	30.9	27.5	64.3
RF_MSSF [46]	90.3	62.7	87.6	80.3	81.8	36.4	92.2	24.1	42.6	56.6
MSDeepVoxNet [39]	88.4	65.3	83.0	67.2	83.8	36.7	92.4	31.3	50.0	78.2
SPG [23]	94.0	73.2	97.4	92.6	87.9	44.0	93.2	31.0	63.5	76.2
GACNet(ours)	91.9	70.8	86.4	77.7	88.5	60.6	94.2	37.3	43.5	77.8

Table 2. Results on the Semantic3D dataset (*reduced-8* challenge).

scribed above.

4. Experiments

In this section, we evaluate the proposed GACNet on various 3D point cloud segmentation benchmarks, including the Stanford Large-Scale 3D Indoor Spaces (S3DIS) [1] dataset and the Semantic3D [15] dataset. The performance is quantitatively evaluated with three metrics, including the per-class intersection over union (IoU), mean IoU of each class (mIoU), and overall accuracy (OA). In addition, the performance of several key components of GAC is further analyzed.

4.1. Indoor segmentation on the S3DIS dataset

The S3DIS dataset contains 3D RGB point clouds from six indoor areas of three different buildings. Each point is annotated with one of the semantic labels from 13 categories. For a principled evaluation, we follow [45, 33, 23] to choose Area 5 as our testing set and train our GACNet on the rest to ensure that the training model does not see any part of the testing area. Notably, Area 5 is not in the same building as other areas and there exist some differences between the objects in Area 5 and other areas. This across-building experimental setup is better for measuring the model’s generalizability, while also brings challenges to the segmentation task.

To prepare our training data, we first split the dataset room by room and then sample them into 1.2m by 1.2m blocks with a 0.1m buffer area on each side. Points lying in the buffer area are regarded as the contextual information and are not linked to the loss function for model training or class prediction. In addition, for training convenience, the points in each block are sampled into a uniform number of 4,096 points. During the testing phase, blocks can be any size depending on the memory of the computing device. In

this experiment, we slice our test room into 3.6m by 3.6m blocks with a maximum of $4,096 \times 9$ points. Each block is individually constructed as a graph pyramid according to Section 3.3 for training or testing.

The quantitative evaluations of the experimental results are provided in Table 1. We can see that the proposed GACNet performs better than other competitive methods in most classes. In particular, we achieve considerable gains in window, table, sofa, and board. In the S3DIS dataset, the board and window are pasted onto the wall and difficult to delineate geometrically, but our GACNet can still segment them out according to their color features. As the convolution weights of GAC are assigned according to not only the spatial positions but also the feature attributes of the neighboring points, the proposed GACNet is able to capture the discriminative features of point clouds even though the spatial geometry is lost or weak.

4.2. Outdoor segmentation on the Semantic3D dataset

The Semantic3D dataset is currently the largest available LiDAR dataset, with over 4 billion points from a variety of urban and rural scenes. Each point has RGB and intensity values and is labeled with one of 8 categories: man-made terrain, natural terrain, high vegetation, low vegetation, buildings, hard scape, scanning artefacts, and cars. Different from the S3DIS dataset, the Semantic3D dataset contains outdoor scenes which have relatively larger objects. To adapt to the size of objects, the sampled blocks for the Semantic3D dataset is set to be 4m by 4m while maintaining the same maximum number of 4,096 points. We provide the evaluation results on the *reduced-8* challenge of the benchmark in Table 2.

Additionally, we list the overall accuracy and mean IoU of our GACNet compared to other state-of-the-art methods.

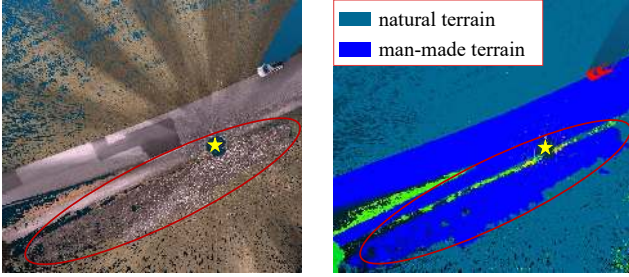


Figure 4. Illustration of an easily-confused area (red ovals) close to the scanning station (yellow stars) and similar to natural terrain in color and geometry but is actually man-made terrain. Note, this kind of area does not appear in our training set and is therefore difficult to segment.

In general, our performance is on par with or better than other competitive methods for many classes. Notably, in the semantic3D dataset, most of the objects, such as cars, hard scape, buildings, and low/high-vegetation, are fragmented and incomplete due to the mutual occlusion among points. However, our GACNet can still learn to capture their discriminative features for segmentation owing to the powerful structured feature learning capability of GAC. Meanwhile, we also notice that the man-made terrain and the natural terrain are relatively difficult to separate for GACNet in this experiment, as there are a large number of points in an easily-confused area (as shown in Figure 4) that does not appear in the training set.

4.3. Ablation studies and analysis

To better understand the influence of various design choices made in the proposed framework, we further conduct several ablation studies to demonstrate the effectiveness of GAC, explore the effect of spatial positions and feature attributes in GAC, compare GAC with CRF-RNN [53], and investigate the effect of initial features.

Effectiveness of GAC. To further understand the effectiveness of the proposed GAC, we compare it with the max operator (including a MLP) of PointNet [33], which has achieved promising results by directly learning on point sets. Specifically, we only replace the attention mechanism in GAC with the max operator while keeping the rest unchanged in our GACNet. The testing results on the S3DIS dataset are provided in Table 3. We can see that the mean IoU of GAC is 4.43% higher than the max operator, which shows that GAC has more advantages in discriminative feature learning than the max operator. Actually, the max operator in PointNet [33] acts as a “max attention” mechanism that tends to characterize the contour of point sets in the feature space while damaging the structural connections between the points of an object. It results in the max operator being good at the object classification task but poor at segmentation where the border of the object needs to be finely delineated.

Ablation studies	OA	mIoU
Max operator	85.47	58.42
Spatial positions only	87.44	60.41
Feature attributes only	87.28	60.25
CRF-RNN (1 iteration)	87.12	61.70
CRF-RNN (3 iterations)	87.86	61.97
CRF-RNN (5 iterations)	87.46	61.83
No RGB	86.06	60.16
No geo-feature	86.17	60.37
Height only	83.56	58.96
GACNet	87.79	62.85

Table 3. Ablation studies on the S3DIS test set.

Spatial positions and feature attributes. In GAC, the neighboring points’ spatial positions and feature attributes serve as spatial and feature guides to dynamically generate their attentional weights. To explore their respective roles, we designed two other variations of GAC that only use the spatial positions and the feature attributes. Their testing results on the S3DIS dataset are reported in Table 3 for comparing convenience. The experimental results show that, both spatial positions and feature attributes have played important roles in GAC for semantic point cloud segmentation. The spatial positions span the unordered neighboring points to meaningful object surfaces, while the feature attributes further guide GAC to adapt to the structure of an object by assigning proper weights to different neighbors. Without the constraint of the spatial positions, points will only exchange information with neighbors with similar initial features, which causes the final features to be piecemeal and difficult to form meaningful objects. Without the guidance of the feature attributes, convolution kernels can hardly distinguish where the object’s border is, and the current points are easily contaminated by the neighboring objects (as shown in Figure 5).

CRF-RNN. As described in Section 3.2, our GACNet actually shares the same characteristics with the CRF model, which encourages feature and label agreement between similar points. To experimentally verify this claim, we remove the last GAC layer in our GACNet and replace it with the CRF-RNN [53] using different iterations. Specifically, we use the Gaussian kernels from [21] for the pairwise potentials of CRF. Their testing results on the S3DIS dataset are also provided in Table 3 for comparing convenience. We can see that, with one iteration the CRF-RNN has basically converged and more iterations do not result in considerably increased accuracy. Since our GACNet has shared the same characteristics of CRF in each layer of the network (Section 3.2), the recurrence of CRF is no longer needed.

Effect of initial features. In the above experiments on the S3DIS dataset, the initial feature vector of each point is composed of height, RGB, and geo-feature. In this sec-

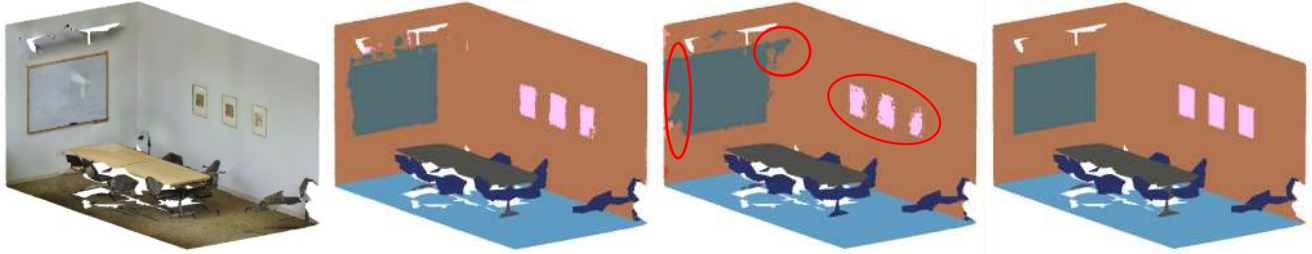


Figure 5. Illustration of the role of feature attributes in GAC. Figures from left to right are the input point cloud, the predicted result by GACNet, the predicted result by GACNet without the feature attributes, and the ground truth. We can see that, with the guidance of the feature attributes, the objects are more clearly and regularly delineated.

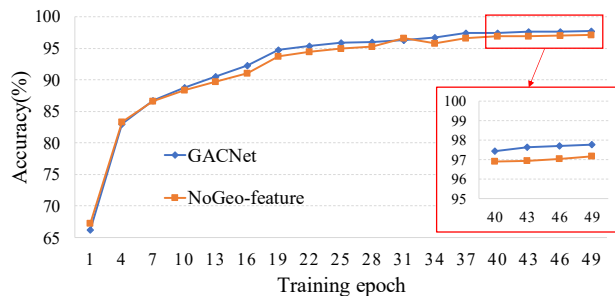


Figure 6. Training accuracy with or without geo-feature.

tion, we provide additional ablation studies to further understand the performance of our GACNet with different initial input features. We design three comparison experiments where we remove the RGB information, the geo-feature, and both of them, respectively. The testing results on the S3DIS dataset are provided in Table 3. By comparison, the mIoU drops by 2.69%, 2.48%, and 3.89% respectively. However, compared to the relatively large accuracy difference in the testing phase, we also notice that the training accuracy without geo-feature actually shows little difference from our standard GACNet (as shown in Figure 6). The initial geo-feature serves as the low-level universal features and is designed according to priori knowledge, which is useful for the generalization of the network.

4.4. Robustness test and stress test

We compare GAC with the max operator [33] on robustness against random Gaussian noise, and resistance on missing data. However, as additional noise will change the class attribute of a point in the segmentation task, we turn to a classification task for our robustness and stress test. We implement this work on the ModelNet40 [50] shape classification benchmark. There are 12,311 CAD models from 40 man-made object categories, splitting them into 9,843 for training and 2,468 for testing. We uniformly sample 1,024 points on their mesh and normalize them into a unit sphere as inputs for our network. Our classification network is built by simply replacing the feature interpolation layers in GACNet with a global pooling layer, and the input of the

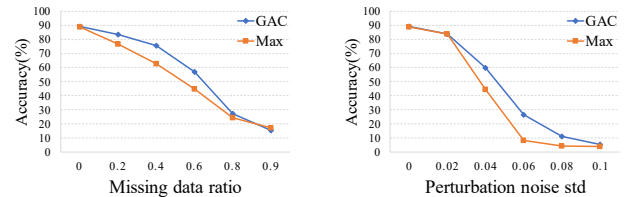


Figure 7. Robustness and stress test. GAC and Max indicate that we use graph attention convolution and the max operator in the classification network respectively.

network is just the height information of each point. All models are trained without data augmentation. During the robustness test, input points are added with Gaussian noise with a series of standard deviations and zero means. For the stress test, a series of ratios of input points are randomly dropped out. From Figure 7, we can see that the max operator is more sensitive to noise because it actually tends to capture the most “special” features (probably noise), while GAC is robust to noise owing to its spatial and feature constraints. For missing data, the accuracy of GAC drops by 13.66% when the missing ratio is 40%, while the max operator drops by 26.48%.

5. Conclusion

We propose a novel graph attention convolution (GAC) with learnable kernel shape for structured feature learning of 3D point cloud. Our GAC is a universal and simple module maintaining the weight sharing property of the standard convolution and can be efficiently implemented on graph data. We have applied GAC to train an end-to-end network for semantic point cloud segmentation. Both theoretical analysis and empirical experiments have shown the effectiveness and advantage of the proposed GAC.

Acknowledgments. This work was supported by the NSFC (No. 41671419, No. 51208392), and the interdisciplinary research program of Wuhan University (No. 2042017kf0204). We are thankful to anonymous reviewers for their helpful comments and suggestions.

References

- [1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D semantic parsing of large-scale indoor spaces. In *CVPR*, pages 1534–1543, 2016.
- [2] Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, and Philip H S Torr. Higher order conditional random fields in deep neural networks. In *ECCV*, 2016.
- [3] James Atwood and Don Towsley. Diffusion-convolutional neural networks. In *NIPS*, 2015.
- [4] Dzmitry Baddanau, KyungHyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.
- [5] Linchao Bao, Yibing Song, Qingxiong Yang, and Narendra Ahuja. An edge-preserving filtering framework for visibility restoration. In *ICPR*, 2012.
- [6] Alexandre Boulch, Joris Guerry, Bertrand Le Saux, and Nicolas Audebert. SnapNet: 3D point cloud semantic labeling with 2D deep segmentation networks. *Computers and Graphics (Pergamon)*, pages 189–198, 2018.
- [7] Michael M. Bronstein, Joan Bruna, Yann Lecun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.
- [8] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. *arXiv:1312.6203*, 2013.
- [9] Liang Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *TPAMI*, 40(4):834–848, 2018.
- [10] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 2016.
- [11] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, and Alán Aspuru-Guzik. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*, 2015.
- [12] Francis Engelmann, Theodora Kontogianni, Alexander Hermans, and Bastian Leibe. Exploring spatial context for 3D semantic segmentation of point clouds. In *ICCV Workshop*, pages 716–724, 2017.
- [13] Jonas Gehring, Michael Auli, David Grangier, and Yann N. Dauphin. A convolutional encoder model for neural machine translation. *arXiv:1611.02344*, 2016.
- [14] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3D semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, 2018.
- [15] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys. Semantic3D.net: A new large-scale point cloud classification benchmark. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2017.
- [16] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017.
- [17] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv:1506.05163*, 2015.
- [18] Evangelos Kalogerakis, Melinos Averkiou, Subhransu Maji, and Siddhartha Chaudhuri. 3D shape segmentation with projective convolutional networks. In *CVPR*, pages 6630–6639, 2017.
- [19] Thomas N Kipf and Max Welling. Semi-supervised classification with graph-concoluational neural networks. In *ICML*, 2017.
- [20] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3D point cloud models. In *CVPR*, pages 863–872, 2017.
- [21] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with gaussian edge potentials. In *NIPS*, pages 109–117, 2012.
- [22] John Lafferty, Andrew McCallum, and Fernando C N Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001.
- [23] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *CVPR*, 2018.
- [24] Truc Le, Giang Bui, and Ye Duan. A multi-view recurrent neural network for 3D mesh segmentation. *Computers and Graphics (Pergamon)*, 2017.
- [25] Truc Le and Ye Duan. PointGrid: A deep network for 3D shape understandings. In *CVPR*, 2018.
- [26] Yangyan Li, Soeren Pirk, Hao Su, Charles R. Qi, and Leonidas J. Guibas. FPNN: Field probing neural networks for 3D data. In *NIPS*, 2016.
- [27] Tsung-Yi Lin, Piotr Doll, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017.
- [28] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen Change Loy, and Xiaoou Tang. Deep learning markov random field for semantic segmentation. *TPAMI*, 40(8):1814–1828, 2018.
- [29] Ping Luo, Xiaogang Wang, and Xiaoou Tang. Pedestrian parsing via deep decompositional network. In *ICCV*, pages 2648–2655, 2013.
- [30] Daniel Maturana and Sebastian Scherer. VoxNet: A 3D convolutional neural network for real-time object recognition. In *IROS*, pages 922–928, 2015.
- [31] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model CNNs. In *CVPR*, pages 5425–5434, 2017.
- [32] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *ICML*, 2016.
- [33] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, pages 77–85, 2017.
- [34] Charles R. Qi, Hao Su, Matthias Niessner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view CNNs for object classification on 3D data. In *CVPR*, pages 5648–5656, 2016.

- [35] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017.
- [36] Xiaojuan Qi, Renjie Liao, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. 3D graph neural networks for RGBD semantic segmentation. In *ICCV*, page 5209–5218, 2017.
- [37] Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Deep learning with sets and point clouds. *arXiv:1611.04500*, 2016.
- [38] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. OctNet: Learning deep 3D representations at high resolutions. In *CVPR*, pages 6620–6629, 2017.
- [39] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. Classification of point cloud scenes with multi-scale voxel deep network. *arXiv:1804.03583*, 2018.
- [40] Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *CVPR*, 2018.
- [41] David I. Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2012.
- [42] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *CVPR*, pages 29–38, 2017.
- [43] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3D shape recognition. In *ICCV*, pages 945–953, 2015.
- [44] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3D. In *CVPR*, 2018.
- [45] Lyne P. Tchapmi, Christopher B. Choy, Iro Armeni, JunYounG Gwak, and Silvio Savarese. SEGCloud: Semantic segmentation of 3D point clouds. In *3DV*, pages 537–547, 2017.
- [46] Hugues Thomas, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Yann Le Gall. Semantic classification of 3D point clouds with multiscale spherical neighborhoods. *arXiv:1808.00495*, 2018.
- [47] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.
- [48] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Dynamic filters in graph convolutional networks. *arXiv:1706.05206*, 2017.
- [49] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-CNN: Octree-based convolutional neural networks for 3D shape analysis. *ACM Transactions on Graphics*, 36(4), 2017.
- [50] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, and Xiaoou Tang. 3D ShapeNets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015.
- [51] Li Yi, Hao Su, Xingwen Guo, and Leonidas Guibas. SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation. In *CVPR*, pages 6584–6592, 2017.
- [52] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets. In *NIPS*, 2017.
- [53] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *ICCV*, pages 1529 – 1537, 2015.