# Boundary-Aware Geometric Encoding for Semantic Segmentation of Point Clouds

**Jingyu Gong,**[1*] **Jiachen Xu,** [1*] **Xin Tan,** [1,3] **Jie Zhou,** [3] **Yanyun Qu,** [4] **Yuan Xie,** [2†] **Lizhuang Ma** [1,2†]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
[2] School of Computer Science and Technology, East China Normal University, Shanghai, China
[3] City University of Hong Kong, HKSAR, China
[4] School of Informatics, Xiamen University, Fujian, China
{gongjingyu, xujiachen, tanxin2017}@sjtu.edu.cn, jzhou67-c@my.cityu.edu.hk
yyqu@xmu.edu.cn, yxie@cs.ecnu.edu.cn, ma-lz@cs.sjtu.edu.cn

## Abstract

Boundary information plays a significant role in 2D image segmentation, while usually being ignored in 3D point cloud segmentation where ambiguous features might be generated in feature extraction, leading to misclassification in the transition area between two objects. In this paper, firstly, we propose a Boundary Prediction Module (BPM) to predict boundary points. Based on the predicted boundary, a boundary-aware Geometric Encoding Module (GEM) is designed to encode geometric information and aggregate features with discrimination in a neighborhood, so that the local features belonging to different categories will not be polluted by each other. To provide extra geometric information for boundary-aware GEM, we also propose a light-weight Geometric Convolution Operation (GCO), making the extracted features more distinguishing. Built upon the boundary-aware GEM, we build our network and test it on benchmarks like ScanNet v2, S3DIS. Results show our methods can significantly improve the baseline and achieve state-of-the-art performance.

## Introduction

Semantic segmentation of point clouds has become an increasingly attended task. Because of the success of 2D image recognition (Long, Shelhamer, and Darrell 2015; Chen et al. 2017), many works tried to extend 2D convolution network to 3D space directly (Maturana and Scherer 2015; Zhou and Tuzel 2018). However, this kind of methods is limited by drastic increment of computational complexity. On the other side, PointNet (Qi et al. 2017a) utilized shared Multi-Layer Perceptrons to directly process point clouds and aggregates information through max-pooling, but it failed to exploit the relationship among points in a local region. Due to the unbalanced distribution of points and irregularity of representation, semantic segmentation of point clouds is still a challenging task.

The boundary plays an important role in the semantic segmentation of point clouds, because lots of misclassifications happen nearby boundary points. In the point cloud, the boundary refers to the transition area between two or more objects belonging to different categories. For example, the

junction of the sofa and the ground can be considered as the boundary. Many works (Wang et al. 2018; Xu et al. 2018; Wu, Qi, and Fuxin 2019) tackled the segmentation problem in point clouds without explicitly learning or using the boundary information, hence they extracted features from points with no differentiation between boundary and non-boundary points. It is noteworthy that extracted features on the boundary are usually ambiguous, because they mix features of points belonging to different categories on different sides of the boundary. As the network goes deeper, if other points incorporate features of the boundary points, these ambiguous features on the boundary will inevitably propagate to more other points hierarchically. So, the information of different objects will spread across the boundary, leading to a bad contour for final semantic segmentation.

To tackle this problem, we propose a Boundary Prediction Module (BPM) to predict boundary points in point clouds. In this module, we give a soft prediction for boundary and this module is skillfully supervised by the ground truth of boundary generated on the fly. It is noteworthy that, compared with semantic segmentation, boundary prediction is easier and likely to obtain better results. So, we introduce the light-weight BPM to predict the boundary. Then, we use the prediction as auxiliary information to boost the performance of segmentation. The BPM and segmentation network are trained jointly in end-to-end manner. Fig. 1 illustrates the predicted boundary in several scenes. Most of them are accurately located between different categories, which also visually reflects the effectiveness of our BPM.

Based upon the BPM, we design a boundary-aware Geometric Encoding Module (GEM) to utilize the predicted boundary in feature extraction. When aggregating local features, we only allow information sharing within each object area by preventing the propagation of features across boundary. Because local features can provide more detail information, mixing local features of different categories will definitely destroy this detail information. Then, in the following layers of encoder where representative points are sampled and global features are encoded, information belonging to different categories can be transferred through boundary to obtain the global scene information. In this way, the predicted boundary would act as a barrier to prevent the information mixture from different categories in local feature ex-

---

Figure 1: Visualization of boundary predicted by our proposed Boundary Prediction Module (BPM). Points in red represent the predicted boundary points. Each visualized scene is selected from ScanNet v2.

traction and be ignored in global feature extraction.

To effectively exploit geometric information, we design a light-weight Geometric Convolution Operation (GCO) which complements geometric features for the boundary-aware GEM. In the GCO, we focus on the angular distribution of neighbors rather than the spatial distribution used in KCNet (Shen et al. 2018) and KPConv (Thomas et al. 2019), which is sensitive to density of points and lack of generalization. In specific, we use a simple vector set as the trainable kernel to learn the geometric pattern. In a neighborhood with $m$ points, its geometric pattern can be represented by $m$ 3-D directional vectors. Therefore, our proposed trainable geometric kernel has the same form. Then, the geometric convolution is a sum over multiplication of vectors in the kernel and directional vectors in the neighborhood. Like the 2D convolution, the response of GCO will be large if the local geometric pattern is similar to the learnt kernel.

Overall, the major contributions can be summarized as follows: (1) We propose a boundary-aware Geometric Encoding Module (GEM) to accurately encode geometric information and prevent the propagation of information across the boundary in local feature extraction. To our best knowledge, we are the first one to take boundary information into the 3D feature aggregation process in an explicit way. (2) The Boundary Prediction Module (BPM), which is supervised with dynamically generated ground truth, is derived to predict the boundary and provide boundary information for the boundary-aware GEM. (3) A Geometric Convolution Operation (GCO) with a learnable vector kernel set is also designed to explore local geometry for each point in a light-weight manner. Experiments on benchmark datasets show that the cutting-edge backbone with the proposed boundary-aware GEM can achieve the state-of-the-art performance.

## Related Work

**Point cloud semantic segmentation.** Intuitively, voxel-based methods (Maturana and Scherer 2015; Zhou and Tuzel 2018) voxelized point clouds and applied 3D grid convolution. Furthermore, SubSparseConv (Graham, Engelcke, and van der Maaten 2018) proposed a convolution for sparse point clouds. However, voxelization inevitably destroys geometric information. PointNet (Qi et al. 2017a) directly ex-

tracted features from the point cloud through shared Multi-Layer Perceptrons. Then, PointNet++ (Qi et al. 2017b) introduced a hierarchical network to aggregate information from a local region and extract features from different scales. But they merely used max-pooling to aggregate information, not typically considering the spatial convolution.

To simulate the spatial convolution operation used in image processing, PCCN (Wang et al. 2018) and Spider-CNN (Xu et al. 2018) utilized MLPs and 3-order function, respectively, to approximate to 3D continuous weight functions w.r.t the position. To deal with the unbalanced distribution of point clouds, PointConv (Wu, Qi, and Fuxin 2019) estimated the density of every point and re-balanced the contribution of each point during convolution according to the point density. PointASNL (Yan et al. 2020) also re-weighted the neighbors to adjust the location of sampled centering point. Additionally, HEPIN (Jiang et al. 2019) introduced an edge branch to exploit the relationship between neighbors and collaborated it with the main branch for fine-grained context information. SPH3D-GCN (Lei, Akhtar, and Mian 2020b) proposed a spherical convolutional kernel splitting the neighborhood into multiple volumetric bins.

All these work tried to simulate the 3D convolution operation aggregating information from a local region without differentiation among points. Compared with these methods, our method is boundary-aware to treat points differently for feature aggregation in a local region, so as to alleviate the propagation of indistinguishable features.

**Boundary in semantic segmentation.** The convolution operation in a local region will aggregate information from neighbors no matter which category they belong to, making the extracted features ambiguous on the boundary, as the neighborhood may include objects belonging to different categories on different sides of boundary. GAC (Wang et al. 2019a) determined the weight of every point's feature according to the similarity, thus alleviating the ambiguity of features introduced by aggregating features of points with different labels. But, no boundary information is explicitly involved, leading to a sub-optimal result. However, the boundary information is quite useful for high-level vision task like semantic segmentation (Bertasius, Shi,

and Torresani 2015). To enhance segmentation coherence, BNF (Bertasius, Shi, and Torresani 2016) used a combination of feature maps to predict the boundary and defined a boundary pairwise potential for energy minimization. BSANet (Zhao et al. 2019) detected the boundary in images and emphasized the features near the boundary at the early stages. These methods proved the importance of boundary for the task of semantic segmentation.

In this work, we also propose the Boundary Prediction Module (BPM) to predict the boundary for point clouds and adjust the feature propagation of boundary points. Compared with BSANet (Zhao et al. 2019), we (1) suppress the propagation of point features on the boundary and (2) predict the boundary for the input point clouds and sample boundary points for other scales.

**Geometric features in point clouds.** Compared with 2D image, point clouds provide more geometric information. ShapeNet (Chang et al. 2015) provided the normal vector along with $xyz$ for every point. PPF-FoldNet (Deng, Birdal, and Ilic 2018) designed geometric features based on the angles between relative positions and normal vectors, but the features themselves were not learnable. KCNet (Shen et al. 2018) proposed to use a learnable point-set kernel to represent the geometric pattern. Specifically, they utilized Gaussian kernel with the distances between kernel points and anchor points as the input to obtain the similarity between point-set kernel and neighbors distribution.

Similary to KCNet, our kernel is also a set of vectors. However, to extract geometric features, our proposed Geometry Convolution Operation (GCO) focuses on the direction rather than the position in the kernel, thus less sensitive to sampling density of points. Besides, our GCO is lightweight and extracts features hierarchically to learn effective geometric patterns, rather than using a heavy-weight module to extract the geometric features just in one layer.

## Methods

First, we will introduce the overall architecture. Second, we will show how we detect the boundary and describe the proposed boundary-aware geometric encoding in detail. Finally, the geometric convolution, which is simply designed but extracts the geometric information efficiently, will be introduced in detail.

### Network Overview

In this paper, we fully consider the geometric characteristics of scenes. Overall, as shown in Fig. 2 (a), we propose an encoder-decoder network composed of a Boundary Prediction Module (BPM) and boundary-aware Geometric Encoding Module (GEM). The BPM is a small and concise neural network to predict the boundary points, so as to provide the boundary cues for boundary-aware GEM to adjust the feature propagation in local regions.

Meanwhile, the boundary-aware GEM also encodes the geometric information of the local region with the help of new derived Geometric Convolution Operation (GCO), that will be described later. It is noteworthy that boundary is only involved when the number of points is large (i.e., the early

stage of encoder and the later stage of decoder). In other layers, all points are treated as the non-boundary points and we only focus on the geometric context.

### Boundary-Aware Geometric Encoding

To implement the boundary-aware GEM, we first introduce a Boundary Prediction Module to predict the boundary points given the point cloud. This module is regularized by the target boundary generated on the fly based on semantic labels. Later, the predicted boundary information is used to impede the propagation of information across the boundary for local feature extraction. By contract, global and abstract features can cross over the boundary to have a better recognition of the global scene.

**Boundary Prediction Module.** First, we automatically annotate each point in training samples as its indicator of the boundary $g$, which is defined in accordance with the label of every point as below. In the target boundary, $g_i$ is $0$ if the $i^{th}$ point is on the boundary, otherwise equal to $1$. For every point $p$, whether it is located on the boundary is determined by its local neighborhood. That is, given fixed number of neighboring points for $p$, if there are more than a predefined ratio (detailed description is in experiments) of points that do not belong to the same category as $p$, then $p$ is assumed to be the point on the boundary, otherwise it is not.

The boundary prediction task is a slightly different from semantic segmentation, as boundary prediction should be aware of the difference of semantic information in a local region. To this end, as shown in Fig. 2. (b), we collect features of $k$ nearest neighbors in the local region for each point and take the variance of collected features as the input of the following part of BPM. Then, like PointNet (Qi et al. 2017a), we utilize several shared MLPs to predict the boundary annotation $\hat{g}$ for the whole input point cloud. Compared with a carefully designed network, our BPM is compact and easy to train. Specifically, its training loss is following:

$$\mathcal{L}_{BPM} = -\sum_{i=1}^{n}(w_1 \cdot g_i \log \hat{g}_i + w_2 \cdot (1-g_i) \log(1-\hat{g}_i)), \quad (1)$$

where $w_1$ and $w_2$ are used to balance the huge difference between the numbers of two categories. We also utilize cross-entropy loss to regularize the final semantic segmentation output, and the total loss is a simple addition of boundary prediction loss and semantic segmentation loss.

**Feature Aggregation with Boundary.** As mentioned above, in the proposed boundary-aware GEM (Fig. 2 (c)), we attempt to block the propagation of local features from points on the boundary in the early stage of encoding process. Therefore, according to the predicted boundary, we utilize the boundary information as a mask/filter to assign different weights to different points during feature aggregation. Before that, we also utilize the GCO (detailed description will be given later) to provide extra geometric features. The main difference between boundary-aware GEM decoder and encoder is that, we do not use the GCO in decoder as the output features of corresponding encoder which have already
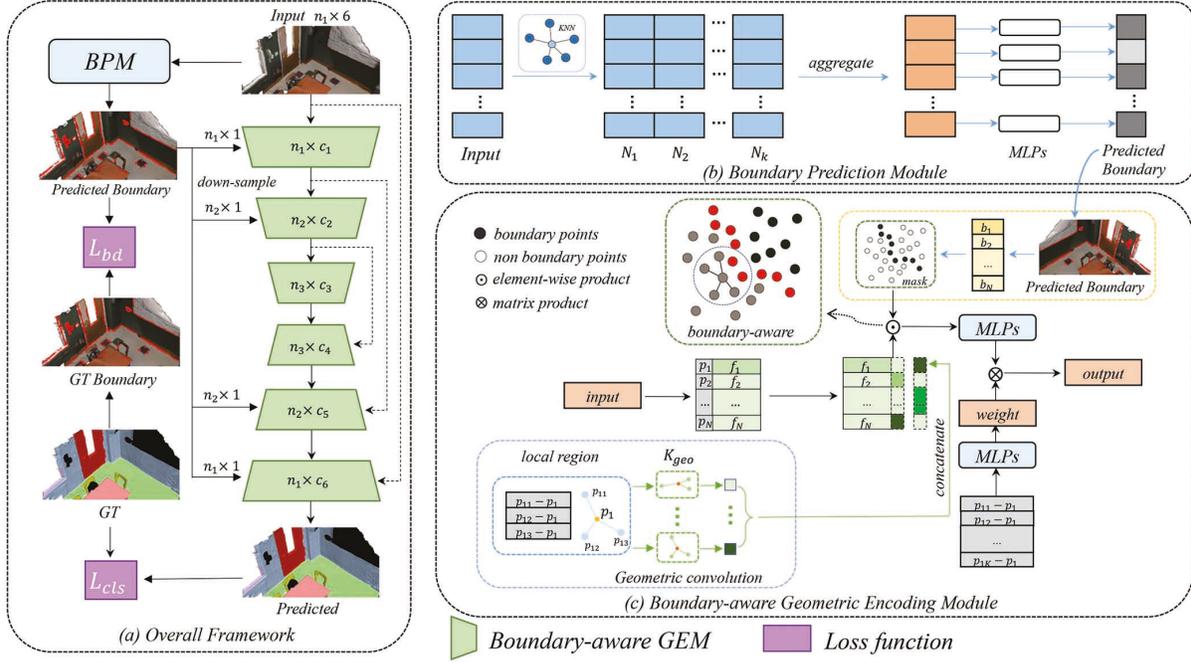
Figure 2: Overall architecture of our network consisting of the Boundary Prediction Module and the boundary-aware Geometric Encoding Module for each layer. (a) illustrates the overall architecture. (b) depicts the architecture of Boundary Prediction Module. (c) describes the boundary-aware Geometric Encoding Module in details.

contained the geometry information will be concatenated to the input of the decoder.

Given the predicted boundary points (the red points in Fig. 2. (c)), during feature aggregation for a grey point, it will collect features in a neighborhood but ignore those points on the boundary. Therefore, the local feature aggregation for point $p_i$ can be expressed as follows:

$$f_{en\_l} = \sigma(\mathcal{A}(\{\phi(r_{ij}) \cdot \mathcal{M}(\hat{g}_j \cdot f_{p_j})\})), \forall p_j \in \mathcal{N}(p_i), \quad (2)$$

where $f_{p_j}$ represents the feature of neighboring $p_j$ containing both original features and geometric features, and $\hat{g}_j$ works as a mask to assign weight to $f_{p_j}$. In this formula, $\mathcal{N}(p_i)$ is the neighborhood of $p_i$, and $\mathcal{M}$ means shared MLPs to combine the original features and extracted geometric features at this scale. Referring to Fig. 2 (c), we can know $\phi$ learns weight from the relative position $r_{ij}$ for neighbor $p_j$ through another few MLPs. Additionally, $\mathcal{A}$ is the aggregation function that is done through matrix product and $\sigma$ represents the activation function. It is noteworthy that $\hat{g}_j$ is 0 if $p_j$ is on the boundary and this boundary point would not contribute to the aggregated feature.

In this way, we prevent the features of points on the boundary to be fused into the extracted local features, thus information is less likely to cross over the boundary to pollute features belonging to other categories (shown in Fig. 2. (c)). We only need to predict the boundary of point clouds for the input layer, while in the later encoding stages, points and predicted boundary labels are down-sampled at the same time. Unlike local features in the first few layers, the global features can propagate among different objects

through boundary points. Therefore, in the latter stage, we extract global features as follows:

$$f_{en\_h} = \sigma(\mathcal{A}(\{\phi(r_{ij}) \cdot \mathcal{M}(f_{p_j})\})), \forall p_j \in \mathcal{N}(p_i). \quad (3)$$

In the decoding stage, the feature extraction procedure is symmetrical. Specifically, when the number of points remains small, global features propagate without impeding to better recognize the global context. While in the later stage of the decoder, we prevent the propagation of features across the boundary again to obtain distinguishing local features.

## Geometric Convolution

To provide extra geometric information for boundary-aware GEM, we propose a light-weight Geometric Convolution Operation (GCO) with a learnable kernel to extract geometric information at different scales, see the bounding box on the left lower corner in Fig. 2 (c).

**Geometric Kernel.** In our method, we propose a geometric kernel $K_{geo}$ with three directional vectors $\{v_1, v_2, v_3 | v_i \in \mathbb{R}^3\}$. Each vector represents a direction in the 3D space, thus the kernel itself can describe a distribution of points over directions, so as to tell where the points is located (e.g., on a plane or curved surface). Unlike (Shen et al. 2018; Thomas et al. 2019) which employ a large amount of kernel points, only three 3-D directional vectors are adopted in our method. Even though the proposed operation has a much simpler structure, the performance is comparable with some sophisticated operators, that is proved in
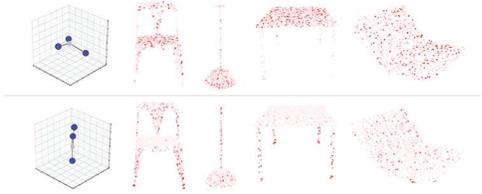
Figure 3: Heat maps for the response of different learnt kernels. The left-most column shows the kernels and the right columns shows the response of some examples (the redder, the larger). These two kernels separately learn the pattern of a horizontal plane and a vertical line.

ablation study. Because, tetrahedron is the simplest polyhedron and these three directional vectors along with the origin can represent a tetrahedron. Furthermore, more complex geometry pattern can be recognized through hierarchical geometric feature extraction. Fig. 3 illustrates learnt kernels and heat maps for different objects to show effectiveness.

**Geometric Convolution Operation.** For a point in the point cloud, the local pattern is represented by the relative positions from this point to its neighbors. Similar to 2D convolution, if the geometric pattern of the neighborhood is very similar to the learnt GCO kernel, the response will be large, thus geometric pattern is recognized.

Our geometric convolution focuses more on the angular distributions of neighbors rather than their relative displacement like KCNet (Shen et al. 2018). For every point $p_i$, relative positions of three neighbors, that are used to represent the local pattern, are represented by $\{\vec{d}_{ij} | \, j \in [1, 2, 3]\}$. Convolving with $K_{geo}$, the output could be expressed as

$$O_i = \max_{\mathcal{P}_i} \sigma(b + \sum_j \vec{d}_{ij} \cdot \vec{v}_{\mathcal{P}_i(j)}), \qquad (4)$$

where $b$ is the bias and $\sigma$ is the activation function. $\mathcal{P}_i(\cdot): \{1, 2, 3\} \mapsto \{1, 2, 3\}$ represents a mapping function which finds the matching vector in the kernel for $\vec{d}_{ij}$. It is noteworthy that because the point cloud is unordered, it is hard to use a fixed mapping. Additionally, if $K_{geo}$ describes the same pattern as the neighborhood, each pair of $\vec{d}_{ij}$ and the matching $\vec{v}_{\mathcal{P}_i(j)}$ would be in the same direction making the dot production maximum. Therefore, in our proposed convolution procedure, we dynamically choose the mapping function that makes the output maximum. Obviously, our geometric convolution is more sensitive to the angle between two vectors $cos\langle\vec{d}_{ij}, \vec{v}_{\mathcal{P}_i(j)}\rangle$ rather than the displacement between vectors in the neighborhood and kernel $|\vec{d}_{ij} - \vec{v}_{\mathcal{P}_i(j)}|$, which is more easily be influenced by scales and density of point clouds.

After extracting geometric features, they are concatenated to the original features of points for further boundary-aware geometric encoding (Fig. 2 (c)), making points with different geometry more distinguishable. In the encoder, geometric patterns can be learnt from different scales, thus complex geometric pattern can be represented by the combination of geometrical features of different scales.

## Experiments

The experiments can be divided into two parts. We demonstrate the performance of our method and compare it with other state-of-the-art methods on ScanNet v2 (Dai et al. 2017) and S3DIS Area-5 (Armeni et al. 2016) for scene semantic segmentation task, respectively. Then, intensive ablation studies are conducted. We take the mean intersection-over-union (mIoU) over categories as our metric like many previous works (Wu, Qi, and Fuxin 2019). Code is available at https://github.com/JchenXu/BoundaryAwareGEM.

### Scene Semantic Segmentation

**Dataset.** In scene semantic segmentation task, we evaluate our method on ScanNet v2 (Dai et al. 2017) and S3DIS (Armeni et al. 2016). In ScanNet v2, there are totally $1,201$ scanned scenes for training and $312$ scenes for validation. Additionally, another 100 scenes are provided as the testing samples, and there are 20 different categories. Following (Wu, Qi, and Fuxin 2019), we randomly sample $3m \times 1.5m \times 1.5m$ cubes from rooms with 8,192 points as the training samples, and test over the entire scan. In S3DIS, there are six indoor areas including 271 rooms from three different buildings. Each point is annotated with a corresponding label from 13 categories. We split points by room and sample all rooms into $0.5m \times 0.5m$ blocks with $0.25m$ padding. Like experiment setting used in previous works (Qi et al. 2017a; Li et al. 2018), we split Area 5 as the test set and use others for training. In the training areas, 4,096 points are sampled for each block and all points in the testing areas are used for testing block-wisely.

**Implementation.** In our method, we take an efficient way to implement the weight computation and feature aggregation using matrix multiplication like PointConv (Wu, Qi, and Fuxin 2019). Therefore, we take PointConv as our baseline, but we do not use density information during feature extraction because it have limited improvement in performance.

In the BPM, to automatically annotate the target boundary points for each input point cloud, points with more than 40% of 32 neighbors not belonging to the same category are assumed to be boundary points. Then, because boundary points are predicted based on neighborhood information, and color information is highly related to boundary prediction, we take the variance of color features of 32 neighbors as the aggregated feature for each point and further predict the boundary points. After predicting boundary points, we build an encoder-decoder network based on the boundary-aware GEM and take both the color and coordinate information as its input. Our model is trained by Adam optimizer with batch size 8 for ScanNet and batch size 12 for S3DIS on a GTX 1080Ti GPU. Also, we analyze the number of the ground truth of boundary and non-boundary points in different scenes. Accordingly, for ScanNet, $w_1$ and $w_2$ used in $\mathcal{L}_{BPM}$ are 1 and 10, and for S3DIS, $w_1$ and $w_2$ are 1 and 2.

**Results.** For ScanNet v2, we report the mean IoU (mIoU) over categories in Table 1, where we have achieved mIoU of 63.5%. It shows our method has outperformed lots of state-of-the-art competitors. Fig. 4 visualizes scene semantic seg-

| Method | mIoU |
| --- | --- |
| PointNet++ (Qi et al. 2017b) | 33.9 |
| PointCNN (Li et al. 2018) | 45.8 |
| 3DMV (Dai and Nießner 2018) | 48.4 |
| PointConv (Wu, Qi, and Fuxin 2019) | 55.6 |
| TextureNet (Huang et al. 2019) | 56.6 |
| HPEIN (Jiang et al. 2019) | 61.8 |
| SegGCN (Lei, Akhtar, and Mian 2020a) | 58.9 |
| SPH3D-GCN (Lei, Akhtar, and Mian 2020b) | 61.0 |
| FusionAwareConv (Zhang et al. 2020) | 63.0 |
| Ours | **63.5** |

Table 1: Semantic segmentation results on ScanNet v2.

| Method | mIoU |
| --- | --- |
| PointNet (Qi et al. 2017a) | 41.09 |
| PointCNN (Li et al. 2018) | 57.26 |
| SPGraph (Landrieu and Simonovsky 2018) | 58.04 |
| PCCN (Wang et al. 2018) | 58.27 |
| ASIS (Wang et al. 2019b) | 53.40 |
| ELGS (Wang, He, and Ma 2019) | 60.06 |
| PAT (Yang et al. 2019) | 60.07 |
| SPH3D-GCN (Lei, Akhtar, and Mian 2020b) | 59.5 |
| GridGCN (Xu et al. 2020) | 57.75 |
| JSNet (Zhao and Tao 2020) | 54.50 |
| Ours | **61.43** |

Table 2: Semantic segmentation results on S3DIS evaluated on Area 5 (Fold #1).

mentation result of PointConv and our method. Misclassification is easy to appear in the transition area of two adjacent objects. For example, in the second row third column, points of "wall" category are predicted as the "picture" that is adjacent to the wall, leading to the poor contour of the picture. By contrast, benefiting from the boundary awareness, our network perform well in this transition area.

For S3DIS, we report the mIoU over categories in Table 2. We achieve $61.43\%$ in mIoU on this benchmark which has better performance than many state-of-the-art competitors. Also, we visualize our segmentation results in Fig. 4. As can be seen in this figure, the segmentation results obtained by our method have better contour thanks to the Boundary-aware GEM for local feature extraction.

## Ablation Study

In this section, we conduct more ablation studies to support our contributions. Because we can only submit one final result to the testing benchmark server of ScanNet, more ablation studies are conducted on the validation set of ScanNet.

**Effectiveness of boundary-aware GEM and GCO.** To show the effectiveness of boundary-aware GEM and the GCO, we conduct more ablative experiments. First, we only use MLPs to simulate the 3D convolution kernel like Point-
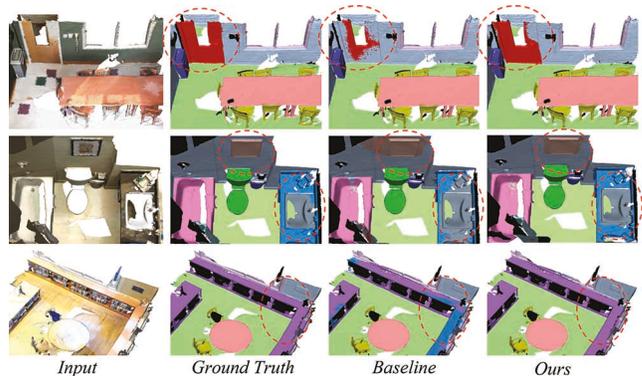


Figure 4: Visualization results of scene semantic segmentation on ScanNet v2. The images from left to right are separately the input scene, ground truth of segmentation, results predicted by PointConv and our method.
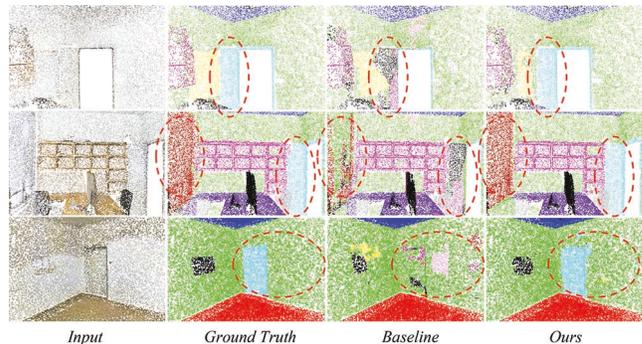


Figure 5: Visualization results of scene semantic segmentation on S3DIS (Area #5). The images from left to right are separately the input scene, ground truth of segmentation, results predicted by PointConv and our method.

Conv (Wu, Qi, and Fuxin 2019) and treat it as our baseline. Next, we simply introduce GCO into the baseline to validate the effectiveness of GCO. Then, we use boundary-aware GEM without GCO to build the network and prove the effectiveness of boundary-aware strategy. Finally, we report the result of our method on validation dataset. The results are shown in Table 3, in which we can see both of them can improve the performance on semantic segmentation task.

**Strategy on Boundary Utilization.** In our method, we attempt to prevent the propagation of features for points on the boundary for local feature extraction. Meanwhile, BSANet (Zhao et al. 2019) proposed to emphasize the features near boundary. So we try to enhance the influence of points on the boundary during local feature aggregation by:

$$f_{en\_l} = \sigma(\mathcal{A}(\{\phi(r_{ij}) \cdot \mathcal{M}((2 - \hat{g}_j) \cdot f_{p_j})\})), \quad (5)$$

where $x_j \in \mathcal{N}(p_i)$ and other settings are the same as our proposed method. Recall the Eq. (2), $\hat{g}_j$ is 0 if $p_j$ is on the boundary. Given the Eq. (5), more emphasis is imposed on boundary points. The result is shown in Table 3 ("Boundary Augmented"), achieving $61.8\%$ in mIoU, which is better

| Method | mIoU |
|---|---|
| Baseline | 58.9 |
| Baseline w/ GCO | 60.4 |
| BAGEM w/o GCO | 60.9 |
| Boundary Augmented | 61.8 |
| Proposed method | **63.4** |

Table 3: Results on the ablation study of boundary-aware GEM. BAGEM represents boundary-aware GEM, Boundary Augmented means to use a strategy to augment the contribution of boundary point in feature aggregation.

| Geo. Encoding | mIoU | FLOPs |
|---|---|---|
| KC | 60.8 | 26.93G |
| KC (H) | 61.0 | 3.87G |
| GCO (2) | 61.1 | 3.65G |
| GCO (6) | 61.7 | 4.20G |
| Proposed method | **63.4** | 3.73G |

Table 4: Results on the ablation study of geometry encoding. KC means replacing GCO with Kernel Correlation (Shen et al. 2018). KC (H) means use a light-weight version of KC hierarchically. The number after GCO means the number of vectors in the kernel.

than not using boundary information. However, compared with our proposed method, it decreases the mIoU by $1.6\%$, that means preventing the propagation of features on the boundary is more effective than emphasizing the features near boundary.

**Performance of GCO compared with KCNet.** Both KCNet and GCO utilize a vector-set kernel to represent localgeometric pattern. Compared with KCNet (Shen et al. 2018), our geometric features are more sensitive to direction rather than position, thus less sensitive to density of points. Additionally, we take a strategy to extract geometric features hierarchically with light-weight geometric convolution to learn complex pattern rather than a heavy-weight module to extract geometric features within one layer.

To show our advantages and give fair comparisons, we first replace GCO with Kernel Correlation (KC) proposed in KCNet with the settings same as KCNet (corresponding to the first row in Table 4). More specifically, the KC is only employed in the first layer with 16 learnable kernel vectors. Moreover, following the settings of our proposed method, we use a light-weight version of KC, that reduces the learnable kernel vectors from 16 to 3, to extract geometric information hierarchically (denoted by KC (H) in Table 4). Also, the extra computational cost is illustrated for these geometry encoding methods in terms of FLOPs. Comparing the row 2 vs. row 1 in Table 4, it is shown that using light-weight version of KC to extract geometric features hierarchically obtains better performance than using heavy-weight KC in

| Method | mIoU |
|---|---|
| No boundary information | 60.4 |
| Random flip | 62.4 |
| Exchange neighboring pair | 61.8 |
| No perturbation | **63.4** |

Table 5: The result of perturbing the predicted boundary point, which shows the robustness of our method.

one layer like KCNet. In addition, using light-weight version of KC decreases the computation drastically. More importantly, keeping other settings the same and using GCO can further increase the mIoU by $2.4\%$. Compared with the light-weight version of KC, GCO has a much simpler form and require less computation resource.

**Number of vectors in geometric kernel.** In our implementation, we utilize a kernel unit with a set of only three 3-D vectors. To check whether a kernel with more or less vectors can learn geometric pattern better, we separately take a kernel with six and two 3-D vectors and other settings are the same. The results are shown in Table 4, where taking six 3-D vectors as kernel achieves $61.7\%$ in mIoU which may be caused by overfitting. It also proves our claim that a kernel with three vectors are enough to learn 3D geometry in a hierarchical way and a kernel with less than three vectors is not able to learn 3D geometry, thus has worse performance.

**Robustness to boundary prediction error.** Furthermore, we also conduct ablation study to show the robustness to prediction error introduced by BPM. First, we randomly flip 3% points on prediction results. We report the results on Table 5 and $62.4\%$ mIoU is achieved on the validation set of ScanNet. We think that 3% is enough because the number of randomly flipped points is comparable to the number of target boundary points. Second, we select 5% of the predicted boundary points and exchange the label of each point with its nearest neighbor, making the boundary points shifted by one point and $61.8\%$ mIoU was achieved. Both outperform the network without predicted boundary, showing the robustness to errors in boundary prediction.

## Conclusion

In this paper, we propose a boundary-geometry aware segmentation method including a Boundary Prediction Module (BPM) and boundary-aware Geometric Encoding Module (boundary-aware GEM) with Geometric Convolution Operation (GCO). The BPM supervised by the dynamically generated target boundary can predict the boundary points in the point cloud. In the boundary-aware GEM, the predicted boundary will guide the feature aggregation by ignoring the contribution of boundary points when collecting features of neighboring points. To exploit the geometry information, we propose the GCO to recognize geometry patterns at different scales and provide extra geometry information. Our proposed method achieves state-of-the-art performance on both ScanNet and S3DIS dataset for 3D semantic segmentation.

## Acknowledgments

## References

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

Bertasius, G.; Shi, J.; and Torresani, L. 2015. High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision. In *Proceedings of the IEEE International Conference on Computer Vision(CVPR)*, 504–512.

Bertasius, G.; Shi, J.; and Torresani, L. 2016. Semantic segmentation with boundary neural fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 3602–3610.

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

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 40(4): 834–848.

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

Dai, A.; and Nießner, M. 2018. 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation. In *Proceedings of the European Conference on Computer Vision(ECCV)*, 452–468.

Deng, H.; Birdal, T.; and Ilic, S. 2018. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *Proceedings of the European Conference on Computer Vision(ECCV)*, 602–618.

Graham, B.; Engelcke, M.; and van der Maaten, L. 2018. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 9224–9232.

Huang, J.; Zhang, H.; Yi, L.; Funkhouser, T.; Nießner, M.; and Guibas, L. J. 2019. Texturenet: Consistent local parametrizations for learning from high-resolution signals on meshes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 4440–4449.

Jiang, L.; Zhao, H.; Liu, S.; Shen, X.; Fu, C.-W.; and Jia, J. 2019. Hierarchical Point-Edge Interaction Network for Point Cloud Semantic Segmentation. In *Proceedings of the IEEE International Conference on Computer Vision(ICCV)*, 10433–10441.

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

Lei, H.; Akhtar, N.; and Mian, A. 2020a. SegGCN: Efficient 3D Point Cloud Segmentation With Fuzzy Spherical Kernel. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR)*, 11611–11620.

Lei, H.; Akhtar, N.; and Mian, A. 2020b. Spherical kernel for efficient graph convolution on 3d point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* .

Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; and Chen, B. 2018. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems(NIPS)*, 820–830.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 3431–3440.

Maturana, D.; and Scherer, S. 2015. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS)*, 922–928. IEEE.

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

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems(NIPS)*, 5099–5108.

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

Thomas, H.; Qi, C. R.; Deschaud, J.-E.; Marcotegui, B.; Goulette, F.; and Guibas, L. J. 2019. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision(ICCV)*, 6411–6420.

Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; and Shan, J. 2019a. Graph Attention Convolution for Point Cloud Semantic Segmentation. In *Proceedings of the IEEE Confer-*

*ence on Computer Vision and Pattern Recognition(CVPR)*, 10296–10305.

Wang, S.; Suo, S.; Ma, W.-C.; Pokrovsky, A.; and Urtasun, R. 2018. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2589–2597.

Wang, X.; He, J.; and Ma, L. 2019. Exploiting Local and Global Structure for Point Cloud Semantic Segmentation with Contextual Point Representations. In *Advances in Neural Information Processing Systems(NIPS)*, 4573–4583.

Wang, X.; Liu, S.; Shen, X.; Shen, C.; and Jia, J. 2019b. Associatively Segmenting Instances and Semantics in Point Clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 4096–4105.

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

Xu, Q.; Sun, X.; Wu, C.-Y.; Wang, P.; and Neumann, U. 2020. Grid-GCN for Fast and Scalable Point Cloud Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 5661–5670.

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

Yan, X.; Zheng, C.; Li, Z.; Wang, S.; and Cui, S. 2020. PointASNL: Robust Point Clouds Processing using Nonlocal Neural Networks with Adaptive Sampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 5589–5598.

Yang, J.; Zhang, Q.; Ni, B.; Li, L.; Liu, J.; Zhou, M.; and Tian, Q. 2019. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 3323–3332.

Zhang, J.; Zhu, C.; Zheng, L.; and Xu, K. 2020. Fusion-Aware Point Convolution for Online Semantic 3D Scene Segmentation. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR)*, 4534–4543.

Zhao, L.; and Tao, W. 2020. JSNet: Joint Instance and Semantic Segmentation of 3D Point Clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 12951–12958.

Zhao, Y.; Li, J.; Zhang, Y.; and Tian, Y. 2019. Multi-class Part Parsing with Joint Boundary-Semantic Awareness. In *Proceedings of the IEEE International Conference on Computer Vision(ICCV)*, 9177–9186.

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