PointASNL: Robust Point Clouds Processing using Nonlocal Neural Networks with Adaptive Sampling

Xu Yan[1,2] Chaoda Zheng[2,3] Zhen Li[1,2,*] Sheng Wang[4] Shuguang Cui[1,2]
[1] The Chinese University of Hong Kong (Shenzhen), [2]Shenzhen Research Institute of Big Data
[3]South China University of Technology, [4]Tencent AI Lab
{xuyan1@link., lizhen@, shuguangcui@}cuhk.edu.cn

Abstract

Raw point clouds data inevitably contains outliers or noise through acquisition from 3D sensors or reconstruction algorithms. In this paper, we present a novel end-to-end network for robust point clouds processing, named PointASNL, which can deal with point clouds with noise effectively. The key component in our approach is the adaptive sampling (AS) module. It first re-weights the neighbors around the initial sampled points from farthest point sampling (FPS), and then adaptively adjusts the sampled points beyond the entire point cloud. Our AS module can not only benefit the feature learning of point clouds, but also ease the biased effect of outliers. To further capture the neighbor and long-range dependencies of the sampled point, we proposed a local-nonlocal (L-NL) module inspired by the nonlocal operation. Such L-NL module enables the learning process insensitive to noise. Extensive experiments verify the robustness and superiority of our approach in point clouds processing tasks regardless of synthesis data, indoor data, and outdoor data with or without noise. Specifically, PointASNL achieves state-of-the-art robust performance for classification and segmentation tasks on all datasets, and significantly outperforms previous methods on real-world outdoor SemanticKITTI dataset with considerate noise. Our code is released through https://github.com/yanx27/PointASNL.

1. Introduction

With the popularity of 3D sensors, it's relatively easy for us to obtain more raw 3D data, e.g., RGB-D data, LiDAR data, and MEMS data [44]. Considering point clouds as the fundamental representative of 3D data, the understanding of point clouds has attracted extensive attention for various applications, e.g., autonomous driving [29], robotics [37], and place recognition [23]. Here, a point cloud has two components: the points $\mathcal{P} \in \mathbb{R}^{N\times 3}$ and the features $\mathcal{F} \in \mathbb{R}^{N\times D}$.

* Corresponding author: Zhen Li.

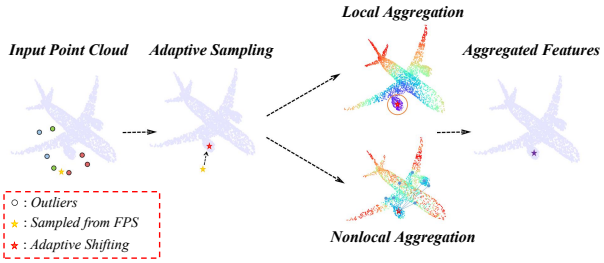


Figure 1. **PointASNL for robust point clouds processing**. The adaptive sampling module adaptively adjusts the sampled point from point clouds with noise. Besides, the local-nonlocal module not only combines the local features in Euclidean space, but also considers the long-range dependency in feature space.

Unlike 2D images, the sparsity and disorder proprieties make robust point clouds processing a challenging task. Furthermore, the raw data obtained from those 3D sensors or reconstruction algorithms inevitably contain outliers or noise in real-world situations.

In this work, we present a novel end-to-end network for robust point clouds processing, named PointASNL, which can deal with point clouds with noise or outliers effectively. Our proposed PointASNL mainly consists of two general modules: adaptive sampling (AS) module and local-nonlocal (L-NL) module. The AS module is used to adjust the coordinates and features of the sampled points, whereas the L-NL module is used to capture the neighbor and long-range dependencies of the sampled points.

Unlike the cases in 2D images, traditional convolution operations cannot directly work on unstructured point cloud data. Thus, most of the current methods usually use sampling approaches to select points from the original point clouds for conducting local feature learning. Among these sampling algorithms, farthest point sampling (FPS) [25], Poisson disk sampling (PDS) [11], and Gumbel subset sampling (GSS) [48] are proposed in previous works. However, as the most representative one, FPS is rooted in Euclidean distance, which is task-dependent and outliers sen-

sitive. PDS, a predefined uniformly sampling method, also cannot solve the problem above in a data-driven way. GSS only performs sampling from a high-dimension embedding space and ignores the spatial distribution of points. Furthermore, the shared key issue in these approaches is that the sampled points are limited to a subset of the original point clouds. Therefore, as shown in the left part of Fig. 1, suppose an outlier point is sampled, it will influence the downstream process inevitably.

To overcome the issues mentioned above, we propose a differentiable adaptive sampling (AS) module to adjust the coordinates of the initial sampled points (e.g., from FPS) via a data-driven way. Such coordinate adjusting facilitates to fit the intrinsic geometry submanifold and further shifts to correct points beyond original point clouds without the influence of outliers. Thus, the AS module can not only benefit point feature learning, but also improve the model robustness to noise.

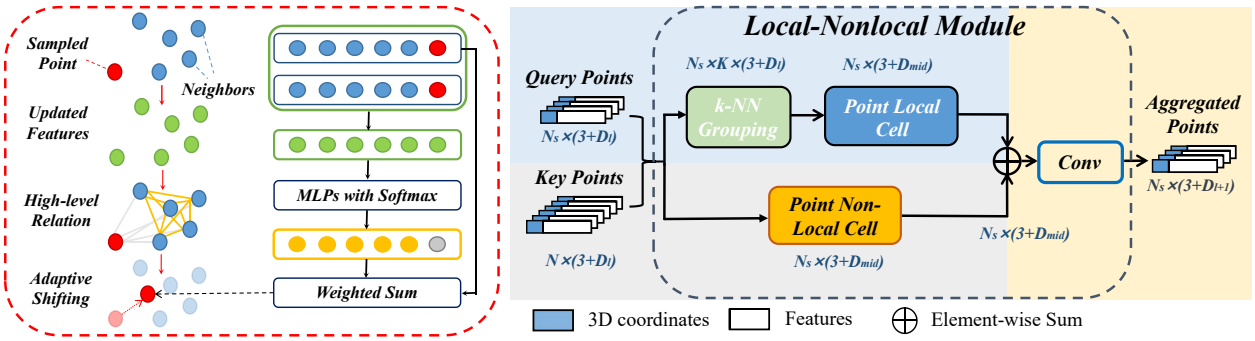
To further enhance the performance as well enables the learning process insensitive to noise, we proposed a local-nonlocal (L-NL) module for capturing neighbor and long-range dependencies of the sampled points. The underlying reason is that, currently, most appealing methods for feature learning is to query a local group around the each sampled point, and then they construct the graph-based learning [30, 42, 50, 14] or define convolution-like operations [12, 47, 8, 3, 44, 34] (we denote them as *Point Local Cell*). Nonetheless, such point local cell only considers local information interaction in the neighbor area and then acquires the global context through a hierarchical structure, which usually leads to bottom-up feature learning. Inspired by the success of the Nonlocal network [41], we innovatively design this L-NL module, in which the key component is the *Point Nonlocal Cell*. In particular, the point nonlocal cell allows the computation of the response of a sampled point as a weighted sum of the influences of the entire point clouds, instead of just within a limited neighbor range. With the learned long-dependency correlation, the L-NL module can provide more precise information for robust point clouds processing. As shown in the right part of Fig. 1, although the sampled points within the lower engine are covered with noise, our L-NL module can still learn the features from the other engine with a different noise distribution.

Our main contribution can be summarized as follows: 1) We propose an end-to-end model for robust point clouds processing, PointASNL, which can effectively ease the influence of outliers or noise; 2) With the proposed adaptive sampling (AS) module, PointASNL can adaptively adjust the coordinates of the initial sampled points, making them more suitable for feature learning with intrinsic geometry and more robust for noisy outliers; and 3) We further design a point nonlocal cell in the proposed local-nonlocal (L-NL) module, which enhances the feature learning in point local cells. Extensive experiments on classification and segmentation tasks verify the robustness of our approach.

2. Related Work

Volumetric-based and Projection-based Methods. Considering the sparsity of point clouds and memory consumption, it is not very effective to directly voxelized point clouds and then use 3D convolution for feature learning. Various subsequent improvement methods have been proposed, e.g., efficient spatial-temporal convolution MinkowskiNet [5], computational effective Submanifold sparse convolution [7], and Oc-tree based neural networks O-CNN [39] and OctNet [27]. Such methods greatly improve the computational efficiency, thus leading to the entire point clouds as input without sampling and superior capacity. There are also other grid-based methods using traditional convolution operations, e.g., projecting 3D data to multi-view 2D images [32] and lattice space [31]. Yet, the convolution operation of these methods lacks the ability to capture nonlocally geometric features.

Point-based Learning Methods. PointNet [24] is the pioneering work directly on sparse and unstructured point clouds, which summarizes global information by using pointwise multi-layer perceptions (MLPs) followed by the max-pooling operation. PointNet++ [25] further applies a hierarchical structure with k-NN grouping followed by max-pooling to capture regional information. Since it aggregates local features simply to the largest activation, regional information is not yet fully utilized. Recently, much effort has been made for effective local feature aggregation. PointCNN [20] transforms neighboring points to the canonical order, which enables traditional convolution to play a normal role. Point2Sequence [21] uses the attention mechanism to aggregate the information of different local regions. Methods [47, 44, 11, 22, 30, 40] directly use the relationship between neighborhoods and local centers to learn a dynamic weight for convolution, where ECC [30] and RS-CNN [22] use ad-hoc defined 6-D and 10-D vectors as edge relationship, PCCN [40] and PointConv [44] project the relative position of two points to a convolution weight. A-CNN [16] uses ring convolution to encode features that have different distances from the local center points, and PointWeb [50] further connects every point pairs in a local region to obtain more representative region features. Still, these methods only focus on local feature aggregation and acquire global context from local features through a hierarchical structure. On the other hand, there are various works for learning the global context from the local features. A-SCN [46] uses a global attention mechanism to aggregate global features but lacks the support of local information, which does not achieve good results. DGCNN [42] proposes the EdgeConv module to generate edge features and search neighbors in



(a) Adaptive Sampling (AS) Module (b) Local-NonLocal Module

Figure 2. Part (a) shows adaptive sampling (AS) module, which firstly updates features of grouping point by reasoning group relationship, then normalized weighs re-weight initial sampled points to achieve new sampled points. Part (b) illustrates the construction of local-nonlocal (L-NL) module, which consists of point local cell and point nonlocal cell. N_s stands for sampled point number, N stands for point number of entire point clouds, D_l, D_{mid}, and D_{l+1} stand for channel numbers.

features space. LPD-Net [23] further extends DGCNN on both spatial neighbors and features neighbors aggregation. Nonetheless, the neighbors in the feature space are not representative of the global features, and spatial receptive fields of the network gradually become confused without a hierarchical structure.

Outlier Removal and Sampling Strategy. Outliers and noise usually exist in raw point clouds data. Previous robust statistics methods [1] for outlier removal suffer from non-trivial parameter tuning or require additional information [43]. Various data-driven methods [9, 26] are proposed for outlier removal, which first discard some outliers and then projects noisy points to clean surfaces. Yet, such methods cannot inherently merge the robust point cloud feature learning with outlier removal in a joint learning manner. On the other hand, deep learning based point cloud processing methods usually sample points to decrease computational consumption. Though, most sampling methods are limited by noise sensitivity and not driven by data [25, 11], or without the consideration of spatial distribution [48]. SO-Net [19] uses an unsupervised neural network, say self-organizing map (SOM), to utilize spatial distribution of point clouds. It then employs PointNet++ [25] to multiple smaller sampled 'nodes'. However, SO-Net does not belong to online adaptive sampling. Under the assumption of local label consistency, some works use the geometric centers of voxel grids to uniformly represent sampled points [34, 30], which ignores the difference of point distribution influence. Still, these methods are extremely sensitive to noise and cannot learn the spatial distribution of sampled points at the same time.

3. Our Method

In this paper, we propose two modules in PointASNL, namely adaptive sampling (AS) module in Sec. 3.1 and local-nonlocal (L-NL) module in Sec. 3.2. In Sec.3.3, we combine AS and L-NL modules in a hierarchical manner to form our proposed PointASNL model.

3.1. Adaptive Sampling (AS) Module

Farthest point sampling (FPS) is widely used in many point cloud framework, as it can generate a relatively uniform sampled points. Therefore, their neighbors can cover all input point clouds as much as possible. Nevertheless, there are two main issues in FPS: (1) It is very sensitive to the outlier points, making it highly unstable for dealing with real-world point clouds data. (2) Sampled points from FPS must be a subset of original point clouds, which makes it challenging to infer the original geometric information if occlusion and missing errors occur during acquisition.

To overcome the above-mentioned issues, we first use FPS to gain the relatively uniform points as original sampled points. Then our proposed AS module adaptively learns shifts for each sampled point. Compared with the similar process widely used in mesh generation [38], the downsampling operation must be taken into account both in spatial and feature space when the number of points is reduced. For the AS module, let $\mathcal{P}_s \in \mathbb{R}^{N_s \times 3}$ as the sampled N_s points from N input points of certain layer, x_i from $\mathcal{P}_s$ and f_i from $\mathcal{F}_s \in \mathbb{R}^{N_s \times D_l}$ as a sampled point and its features. We first search neighbors of sampled points as groups via k-NN query, then use general self-attention mechanism [35] for group features updating.

As shown in Fig. 2 (a), we update group features by using attention within all group members. For $x_{i,1}, ..., x_{i,K} \in \mathcal{N}(x_i)$ and their corresponding features $f_{i,1}, ..., f_{i,K}$, where $\mathcal{N}(x_i)$ is K nearest neighbors of sampled point x_i, feature updating of group member $x_{i,k}$ can be written as

$$f_{i,k} = \mathcal{A}(\mathcal{R}(x_{i,k}, x_{i,j})\gamma(x_{i,j}), \quad \forall x_{i,j} \in \mathcal{N}(x_i)), \qquad (1)$$

where a pairwise function $\mathcal{R}$ computes a high level relationship between group members $x_{i,k}, x_{i,j} \in \mathcal{N}(x_i)$. The

unary function γ change the each group feature $f_{i,j}$ from dimension D_l to another hidden dimension D' and $\mathcal{A}$ is a aggregation function.

For less computation, we consider γ in the form of a linear transformation of point features $\gamma(x_{i,j}) = W_\gamma f_{i,j}$, and relationship function $\mathcal{R}$ is dot-product similarity of two points as follows,

$$\mathcal{R}(x_{i,k}, x_{i,j}) = \text{Softmax}(\phi(f_{i,k})^T \theta(f_{i,j})/\sqrt{D'}), \quad (2)$$

where ϕ and θ are independent two linear transformations and can be easily implemented by independent 1D convolution $Conv : \mathbb{R}^{D_l} \mapsto \mathbb{R}^{D'}$, where D_l and D' are input and output channel, respectively.

After that, point-wise MLPs, i.e., σ_p and σ_f with softmax activation function on K group members are used to obtain the corresponding intensity of each point in a group, which can be represented as normalized weights for each coordinate axis and features channel.

$$\begin{aligned} F_p &= \{\sigma_p(f_{i,k})\}_{k=1}^K, \quad W_p = \text{Softmax}(F_p), \\ F_f &= \{\sigma_f(f_{i,k})\}_{k=1}^K, \quad W_f = \text{Softmax}(F_f), \end{aligned} \quad (3)$$

where $F_p, F_f, W_p, W_f \in \mathbb{R}^{K\times 1}$ are outputs of point-wise MLPs and normalized weights after softmax function. Finally, a adaptive shifting on both K neighbors' coordinates from $X \in \mathbb{R}^{K\times 3}$ and their features from $F \in \mathbb{R}^{K\times D'}$ are implemented by the weighted sum operation. We obtain a new coordinate of the sampled point x_i^* and its features f_i^* by following operations,

$$\begin{aligned} x_i^* &= W_p^T X, \quad X = \{x_{i,k}\}_{k=1}^K, \\ f_i^* &= W_f^T F, \quad F = \{f_{i,k}\}_{k=1}^K. \end{aligned} \quad (4)$$

3.2. Local-Nonlocal (L-NL) Module

Within our L-NL module, there are two cells: point local (PL) cell and point nonlocal (PNL) cell. Specifically, the PL cell can be any appealing algorithms (e.g., PointNet++ [25], PointConv [44]), and the PNL cell innovatively considers the correlations between sampled points and the entire point cloud in multi-scale. Consequently, the contextual learning of the point cloud is enhanced by combining the local and global information (See Fig. 2(b)).

3.2.1 Point Local Cell

The local features mining of point clouds often exploits the local-to-global strategy [25], which aggregates local features in each group and gradually increases the receptive field by hierarchical architectures. We adopt such methods in point local (PL) cell. Similar to the previous definition for a local sampled point x_i, corresponding feature f_i and neighborhoods $\mathcal{N}(x_i)$, a generalized local aggregation function used in PL can be formulated as

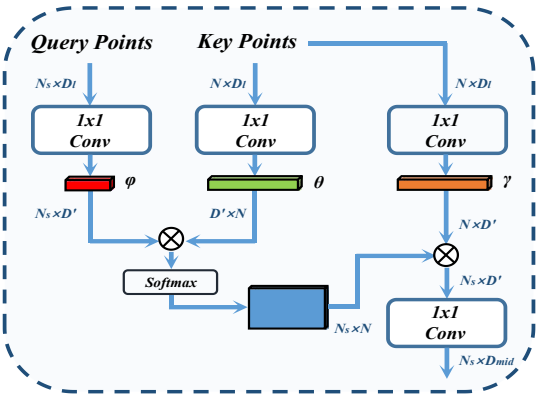


Figure 3. **Inner structure of point nonlocal (PNL) cell.** For the notations N_s, N, D_l, D_{mid} please refer to the caption of Fig. 2, D' is the intermediate channel numbers.

$$f_i^l = \mathcal{A}(\mathcal{L}(f_n), \forall x_n \in \mathcal{N}(x_i)), \quad (5)$$

where f_i^l is updated features of local center x_i, which is updated by local feature transformation function $\mathcal{L}$ and aggregation function $\mathcal{A}$. For PointNet++ [25], $\mathcal{L}$ is multi-layer perceptions (MLPs) and $\mathcal{A}$ is max-pooling. Recently, more and more works directly design convolution operators on the local regions, which mainly change $\mathcal{L}$ to be a learnable weighted multiply obtained by neighbor relationships. Considering the efficiency and effectiveness of the operation in a compromise, we implement the convolution operation by adaptively projecting the relative position of two points to a convolution weight [40, 44], and aggregate local features,

$$\mathcal{L}(f_n) := g(x_n - x_i) f_n, \quad (6)$$

where g is chosen as MLPs: $\mathbb{R}^3 \mapsto \mathbb{R}^{D_l \times D_{mid}}$, which transfers 3 dimension relative position to $D_l \times D_{mid}$ transformation matrix. D_l represents the channel of the input features in certain layer and D_{mid} is the channel of the updated features by PL cell.

3.2.2 Point Nonlocal Cell

Inspired by nonlocal neural networks [41] in 2D images for long-range dependencies learning, we design a specific point nonlocal (PNL) cell for global context aggregation (Fig. 3). There are two main differences between our point nonlocal cell and component proposed in [41]: (1) We use our sampled points as query points to calculate similarity with entire points in certain layers (say, key points $\mathcal{P}_k$). Furthermore, our query points are not limited within a subset of input point clouds, as each sampled point adaptively updates its coordinate and features by the AS module (Sec. 3.1). (2) Our output channel is gradually increased with the down-sampling operation in each layer,

which avoids information loss in the down-sampling encoder. Specifically, similar with Eq. 1, given query point x_i and key point from $\mathcal{P}_\text{k}$, the nonlocal operation $\mathcal{NL}$ is defined as:

$$\mathcal{NL}(x_i, \mathcal{P}_\text{k}) := \mathcal{A}(\mathcal{R}(f_i, f_j)\gamma(f_j), \forall x_j \in \mathcal{P}_\text{k}), \quad (7)$$

where $\mathcal{P}_k \in \mathbb{R}^{N\times 3}$ stands for the entire N key points in a certain layer. Finally, a single nonlinear convolution layer σ fuse the global context and adjust the channel of each point to the same dimension with the output of PL D_{l+1} (Eq. 5). Hence, for a sampled point x_i, its updated feature is computed by PNL with function

$$f_i^{nl} = \sigma(\mathcal{NL}(x_i, \mathcal{P}_k)). \quad (8)$$

3.2.3 Local-Nonlocal (L-NL) Fusion

By combining PL and PNL, we construct a local-nonlocal module to encode local and global features simultaneously. As shown in Fig. 2 (b), it uses query points and key points as inputs, and exploit k-NN grouping for neighborhoods searching for each query point. Then, the group coordinates and features of each local region are sent through PL for local context encoding. For PNL, it uses whole key points to integrate global information for each query point via an attention mechanism. Finally, for each updated point, a channel-wise sum with a nonlinear convolution σ is used to fuse local and global information.

3.3. PointASNL

By combining the two components proposed in Sec.3.1 and Sec 3.2 in each layer, we can implement a hierarchical architecture for both classification and segmentation tasks.

For the classification, we designed a three-layer network and down-sample input points at two levels. In particular, the first two layers sample 512 and 124 points. The third layer concatenate global features of former two layers with max pooling, where new features are processed by fully connected layers, dropout, and softmax layer, respectively. The batch normalization layers and the ReLU function are used in each layer. Furthermore, skip connections [10] are used in the first two layers.

For the segmentation (see Fig. 4), each encoder layer is similar with the setting in classification, but network has a deeper structure (1024-256-64-16). In the decoder part, we use 3-nearest interpolation [25] to get the up-sampled features and also use the L-NL Block for better feature learning. Furthermore, skip connections are used to pass the features between intermediate layers of the encoder and the decoder.

4. Experiment

We evaluate our PointASNL on various tasks, including synthetic dataset, large-scale indoor and outdoor scene segmentation dataset. In all experiments, we implement the models with Tensorflow on one GTX 1080Ti GPU.

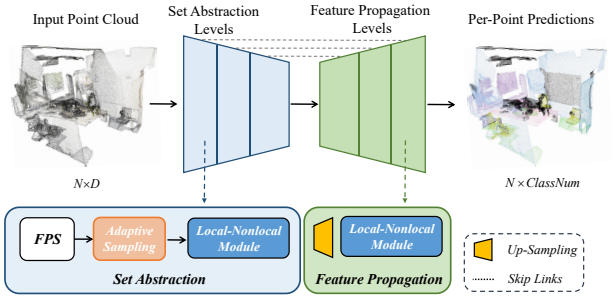


Figure 4. Architecture of our PointASNL for point cloud semantic segmentation. The L-NL modules are used in both encoder and decoder.

4.1. Classification

We evaluate our model on synthetic dataset *ModelNet10* and *ModelNet40* [45] for classification, where *ModelNet40* is composed of 9843 train models and 2468 test models in 40 classes and *ModelNet10* is a subset of *ModelNet40* that consists of 10 classes with 3991 training and 908 testing objects.

Shape Classification. Training and testing data in classification are provided by [24]. For training, we select 1024 points as the input. The augmentation strategy includes the following components: random anisotropic scaling in range $[-0.8, 1.25]$, translation in the range $[-0.1, 0.1]$, and random dropout 20% points. For testing, similar to [24, 25], we apply voting test using random scaling and then average the predictions. In Tab. 1, our method outperforms almost all state-of-the-art methods in 1024 input points except RS-CNN. Note that RS-CNN [22] can achieve 93.6% from 92.9% on uniformly sampling with tricky voting strategy (the best of 300 repeated tests), which is different from normal random sampling and once voting setting.

Shape Classification with Noise. Most of the methods can achieve decent performance on synthetic datasets, as they have stable distribution and do not contain any noise. Though, such a good performance often leads to a lack of robustness of the model. To further verify the robustness of our model, we did the experiments like KC-Net [28] to replace a certain number of randomly picked points with random noise ranging $[-1.0, 1.0]$ during testing. The comparisons with PointNet [24], PointConv [44] and KC-Net [28] are shown in Fig. 5 (b). As shown in this figure, our model is very robust to noise, especially after adding the AS module. It can be seen from (c) and (d) that the adaptive sampling guarantees the proper shape of the sampled point clouds, making the model more robust.

Table 1. Overall accuracy on ModelNet10 (M10) and ModelNet40 (M40) datasets. "pnt" stands for coordinates of point and "nor" stands for normal vector.

Method	input	#points	*M10*	*M40*
O-CNN [39]	pnt, nor	-	-	90.6
SO-Net [19]	pnt, nor	2k	94.1	90.9
Kd-Net [15]	pnt	32k	94.0	91.8
PointNet++ [25]	pnt, nor	5k	-	91.9
SpiderCNN [47]	pnt, nor	5k	-	92.4
KPConv [34]	pnt	7k	-	92.9
SO-Net [19]	pnt, nor	5k	**95.7**	**93.4**
Pointwise CNN [12]	pnt	1k	-	86.1
ECC [30]	graphs	1k	90.8	87.4
PointNet [24]	pnt	1k	-	89.2
PAT [48]	pnt, nor	1k	-	91.7
Spec-GCN [36]	pnt	1k	-	91.8
PointGrid [18]	pnt	1k	-	92.0
PointCNN [20]	pnt	1k	-	92.2
DGCNN [42]	pnt	1k	-	92.2
PCNN [3]	pnt	1k	94.9	92.3
PointConv [44]	pnt, nor	1k	-	92.5
A-CNN [16]	pnt, nor	1k	95.5	92.6
Point2Sequence [21]	pnt	1k	95.3	92.6
RS-CNN [22]	pnt	1k	-	93.6
PointASNL	pnt	1k	95.7	92.9
PointASNL	pnt, nor	1k	**95.9**	**93.2**

4.2. Segmentation

Indoor Scene Segmentation.[1] Unlike classification on synthetic datasets [45, 49], indoor 3D scene segmentation is a more difficult task, because it is real-world point clouds and contains lots of outliers and noise. We use *Stanford 3D Large-Scale Indoor Spaces (S3DIS)* [2] and *ScanNet v2* (*ScanNet*) [6] datasets to evaluate our model.

S3DIS dataset is sampled from 3 different buildings, which includes 6 large-scale indoor areas with 271 rooms. Each point in this dataset has a semantic label that belongs to one of the 13 categories. We compare mean per-class IoU (mIoU) on both 6-fold cross-validation over all six areas and Area 5 (see supplementary material). *ScanNet* dataset contains 1513 scanned indoor point clouds for training and 100 test scans with all semantic labels unavailable. Each point has been labeled with one of the 21 categories. We submitted our results to the official evaluation server to compare against other state-of-the-art methods on the benchmark.

During the training process, we generate training data by randomly sample $1.5m \times 1.5m \times 3m$ cubes with 8192 points from the indoor rooms. $0.1m$ padding of sampled cubes is used to increase the stability of the cube edge prediction, which is not considered in the loss calculation. On both datasets, we use points position and RGB information as features. We did not use the relative position in PointNet [24] as a feature to train the model in *S3DIS*, because our model already learns relative position information well. During the evaluation process, we use a sliding window over the entire rooms with $0.5m$ stride to complement 5 voting test.

[1]Supplementary material shows that with more sampled points and deeper structure, our PointASNL can still achieve further improvement to 66.6% on *ScanNet* benchmark.

Table 2. Segmentation results on indoor S3DIS and ScanNet datasets in mean per-class IoU (mIoU,%).

Method	S3DIS	ScanNet
methods use unspecific number of points as input		
TangentConv [33]	52.8	40.9
SPGraph [17]	62.1	-
KPConv [34]	**70.6**	**68.4**
methods use fixed number of points as input		
PointNet++ [25]	53.4	33.9
DGCNN [42]	56.1	-
RSNet [13]	56.5	-
PAT [48]	64.3	-
PointCNN [20]	65.4	45.8
PointWeb [50]	66.7	-
PointConv [44]	-	55.6
HPEIN [14]	67.8	61.8
PointASNL	**68.7**	**63.0**

In Tab. 2, we compare our PointASNL with other state-of-the-art methods under the same training and testing strategy (randomly chopping cubes with a fixed number of points), e.g., PointNet++ [25], PointCNN [20], PointConv [44], PointWeb [50] and HPEIN [14]. We also list results of another kind of methods (using points of unfixed number or entire scene as input), e.g., TangentConv [33]

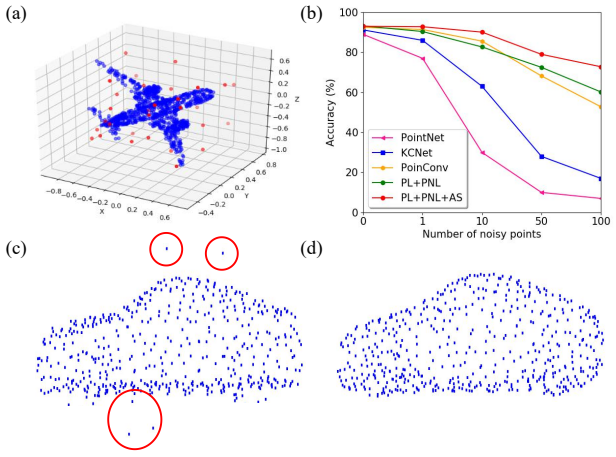


Figure 5. (a) Point cloud with some points being replaced with random noise. (b) Classification results of different models with noisy points, where PL, PNL, AS mean point local cell, point nonlocal cell and adaptive sampling, respectively. (c) Farthest point sampling on noisy data. (d) Adaptive sampling on noisy data, which maintain the distribution of the point cloud.

(a) S3DIS *(b) ScanNet*

Figure 6. Examples of indoor semantic segmentation on *S3DIS* and *ScanNet* datasets.

Table 3. Semantic Segmentation results on SemanticKITTI, where "pnt" stands for coordinates of point and SPGraph [17] uses all of points as input.

Method	input	mIoU(%)
PointNet [24]	50k pnt	14.6
SPGraph [17]	-	17.4
SPLATNet [31]	50k pnt	18.4
PointNet++ [25]	45k pnt	20.1
TangentConv [33]	120k pnt	40.9
PointASNL	8k pnt	**46.8**

and KPconv [34]. All methods use only point clouds as input without voxelization.

As shown in Tab. 2, PointASNL outperforms all methods with same the training strategy in both *S3DIS* and *ScanNet*. In particular, our result is 8% higher than previous state-of-the-art PointConv [44] on the ScanNet with the same experiment setting, in which the convolution design is similar to our PL cell. Nevertheless, without proper sampling and global information support, it cannot achieve such results with the same network architecture.

On the other hand, training using more points as input can obtain more information. Instead of learning from randomly selected cubes with fixed number, KP-Conv [34] performs grid sampling based on the assumption of local label consistency so that larger shape of the point cloud can be included as input.

The qualitative results are visualized in Fig. 6. Our method can correctly segments objects even in complex scenes.

Outdoor Scene Segmentation. Compared with its indoor counterpart, an outdoor point cloud covers a wider area and has a relatively sparser point distribution with noise. For this reason, it is more challenging to inference from outdoor scenes.

We evaluated our model on SemanticKITTI [4], which is a large-scale outdoor scene dataset, including 43,552 scans captured in the wild. The dataset consists of 22 sequences (00 to 10 as the training set, and 11 to 21 as the test set), each of which contains a series of sequential laser-scans. Each individual scan is a point clouds generated with a commonly used automotive LiDAR. The whole sequence can be generated by aggregating multiple consecutive scans.

In our experiments, we only evaluated our model under a single scan semantic segmentation. In the single scan experiment [4], sequential relations among scans in the same sequence are not considered. The total number of 19 classes is used for training and evaluation. Specifically, the input data generated from the scan is a list of coordinates of the three-dimensional points along with their remissions.

During training and testing, we use a similar sliding windows based strategy as indoor segmentation. Since point clouds in the outdoor scene are more sparse, we set the size of the cube with $10m \times 10m \times 6m$ and $1m$ padding. In Tab. 3, we compare PointASNL with other state-of-the-art methods. Our approach outperforms others by a large margin. Supplementary material shows that our method achieves the best result in 13 of 19 categories. Furthermore, Fig. 7 illustrates our qualitative visualization of two samples, even if the scene is covered with a lot of noise

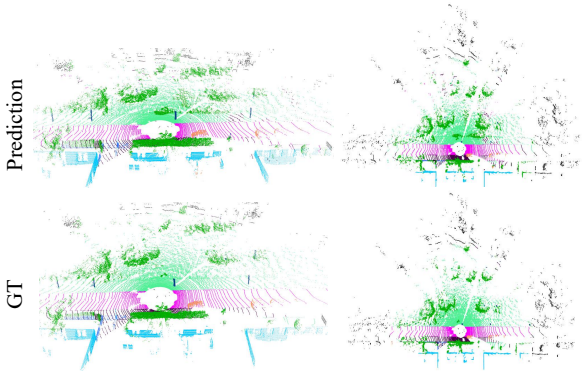


Figure 7. Example of outdoor SemanticKITTI datasets.

Table 4. Ablation study on *ModelNet40* and *ScanNet v2* validation set. PL, PNL and AS mean point local cell, point nonlocal cell and adaptive sampling.

Model	Ablation	*ModelNet40*	*ScanNet*
A	PNL only	90.1	45.7
B	PL only	92.0	56.1
C	PL+PNL	93.2	60.8
D	PL+PNL+AS	**93.2**	**63.5**
E	PointNet2 [24]	90.9	48.9
F	PointNet2+PNL	93.0	54.6
G	PointNet2+PNL+AS	92.8	55.4
H	DGCNN [42]	92.2	52.7
I	DGCNN+PNL	92.9	56.7
J	DGCNN+PNL+AS	93.1	58.3

caused by unmanned collection, our model can still predict perfectly.

4.3. Ablation Study

To further illustrate the effectiveness of proposed AS and L-NL module, we designed an ablation study on both the shape classification and the semantic segmentation. The results of the ablation study are summarized in Tab. 4.

We set two baselines: A and B. Model A only encodes global features by PNL, and model B only encodes local features. The baseline model A gets a low accuracy of 90.1% and 45.7% IoU on segmentation, and model B gets 92.0% and 56.1%, respectively. When we combine local and global information (models C), there is a notable improvement in both classification and segmentation. Finally, when we add the AS module, the model will have a significant improvement in the segmentation task (93.2% and 63.5% in model D).

Furthermore, our proposed components L-NL module and AS module can directly improve the performance of other architecture. When we use PointNet++ [25] in our PL cell (model F), it will reduce the error of classification and segmentation tasks by 23.1% and 12.6%, respectively, with its original model (model E). It should be noted that the AS module does not increase the accuracy of the classification task, even reduced the accuracy of classification when adding on PointNet++ (model F). This is because the synthetic dataset does not have a lot of noise like scene segmentation, for some simpler local aggregation (e.g., max pool), it may make them unable to adapt the uneven point cloud distribution after using AS. Furthermore, we also use DGCNN [42] as our local aggregation baseline (model H), and fused architecture (model I and J) can largely improve the performance on two datasets.

4.4. Robustness for Sparser Point Clouds

To further verify the robustness of the PointASNL model, we take sparser points (i.e., 1024, 512, 256, 128 and 64) as the input to various models trained with 1024 points. Then we compare our method with PointNet [24], PointNet++ [25], SO-Net [19] and the recent state-of-the-art RS-CNN [22]. We follow these methods to apply random input dropout during the training.

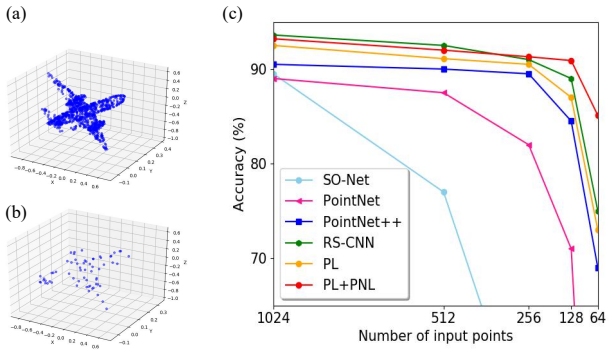


Figure 8. (a) A sample of input point cloud. (b) A sample of input point cloud after randomly select 64 points (c) Results of testing with sparser points.

As can be seen from the Fig .8 (c), PNL can greatly improve the robustness of our model with different density inputs. In particular, when the input contains only 64 points, PNL can even help to improve the accuracy of our model, from 73.9% to 85.2%, which largely exceeds the current state-of-the-art RS-CNN [22] (about 75%). The experimental results fully demonstrate that the use of local and global learning methods can greatly improve the robustness of the model. As shown in Fig .8 (a) and (b), when the input points reduce to 64, even humans can hardly recognize the airplane, but our model can classify it correctly. Such superior robustness makes our proposed PointASNL model suitable for raw noisy point clouds with limited sampling points, especially for large scale outdoor scenario.

5. Conclusion

We have presented the adaptive sampling (AS) and the local-nonlocal (L-NL) module to construct the architecture of PointASNL for robust 3D point cloud processing. By combining local neighbors and global context interaction, we improve traditional methods dramatically on several benchmarks. Furthermore, adaptive sampling is a differentiable sampling strategy to fine-tune the spatial distribution of sampled points, largely improve the robustness of the network. Experiments with our state-of-the-art results on competitive datasets and further analysis illustrate the effectiveness and rationality of our PointASNL.

Acknowledgment. The work was supported by grants No. 2018YFB1800800, NSFC-61902335, No. 2019E0012, No. ZDSYS201707251409055, No. 2017ZT07X152, No. 2018B030338001, and CCF-Tencent Open Fund.

References

[1] Charu C Aggarwal. Outlier analysis. In *Data mining*, pages 237–263. Springer, 2015. 3

[2] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1534–1543, 2016. 6

[3] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *arXiv preprint arXiv:1803.10091*, 2018. 2, 6

[4] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019. 7

[5] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 2

[6] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. 6

[7] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9224–9232, 2018. 2

[8] Fabian Groh, Patrick Wieschollek, and Hendrik PA Lensch. Flex-convolution. In *Asian Conference on Computer Vision*, pages 105–122. Springer, 2018. 2

[9] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J Mitra. Pcpnet learning local shape properties from raw point clouds. In *Computer Graphics Forum*, volume 37, pages 75–85. Wiley Online Library, 2018. 3

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[11] Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. In *SIGGRAPH Asia 2018 Technical Papers*, page 235. ACM, 2018. 1, 2, 3

[12] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 984–993, 2018. 2, 6

[13] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2626–2635, 2018. 6

[14] Li Jiang, Hengshuang Zhao, Shu Liu, Xiaoyong Shen, Chi-Wing Fu, and Jiaya Jia. Hierarchical point-edge interaction network for point cloud semantic segmentation. 2019. 2, 6

[15] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 863–872, 2017. 6

[16] Artem Komarichev, Zichun Zhong, and Jing Hua. A-cnn: Annularly convolutional neural networks on point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7421–7430, 2019. 2, 6

[17] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4558–4567, 2018. 6, 7

[18] Truc Le and Ye Duan. Pointgrid: A deep network for 3d shape understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9204–9214, 2018. 6

[19] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9397–9406, 2018. 3, 6, 8

[20] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 820–830, 2018. 2, 6

[21] Xinhai Liu, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8778–8785, 2019. 2, 6

[22] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019. 2, 5, 6, 8

[23] Zhe Liu, Shunbo Zhou, Chuanzhe Suo, Peng Yin, Wen Chen, Hesheng Wang, Haoang Li, and Yun-Hui Liu. Lpd-net: 3d point cloud learning for large-scale place recognition and environment analysis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2831–2840, 2019. 1, 3

[24] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017. 2, 5, 6, 7, 8

[25] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 1, 2, 3, 4, 5, 6, 7, 8

[26] Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J Mitra, and Maks Ovsjanikov. Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In *Computer Graphics Forum*. Wiley Online Library, 2019. 3

[27] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017. 2

[28] Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4548–4557, 2018. 5

[29] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 1

[30] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3693–3702, 2017. 2, 3, 6

[31] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018. 2, 7

[32] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proc. IEEE Int. Conf. Comput. Vision*, pages 945–953, 2015. 2

[33] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3887–3896, 2018. 6, 7

[34] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. *arXiv preprint arXiv:1904.08889*, 2019. 2, 3, 6, 7

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3

[36] Chu Wang, Babak Samari, and Kaleem Siddiqi. Local spectral graph convolution for point set feature learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–66, 2018. 6

[37] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In *Robotics: Science and Systems*, volume 1, pages 10–15607, 2015. 1

[38] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 3

[39] 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 (TOG)*, 36(4):72, 2017. 2, 6

[40] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2589–2597, 2018. 2, 4

[41] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018. 2, 4

[42] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 2, 6, 8

[43] Katja Wolff, Changil Kim, Henning Zimmer, Christopher Schroers, Mario Botsch, Olga Sorkine-Hornung, and Alexander Sorkine-Hornung. Point cloud noise and outlier removal for image-based 3d reconstruction. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 118–127. IEEE, 2016. 3

[44] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019. 1, 2, 4, 5, 6, 7

[45] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 5, 6

[46] Saining Xie, Sainan Liu, Zeyu Chen, and Zhuowen Tu. Attentional shapecontextnet for point cloud recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4606–4615, 2018. 2

[47] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 87–102, 2018. 2, 6

[48] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3323–3332, 2019. 1, 3, 6

[49] Li Yi, Vladimir G Kim, Duygu Ceylan, I Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, Leonidas Guibas, et al. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (TOG)*, 35(6):210, 2016. 6

[50] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5565–5573, 2019. 2, 6