

# Self-positioning Point-based Transformer for Point Cloud Understanding

Jinyoung Park<sup>1\*</sup>, Sanghyeok Lee<sup>1\*</sup>, Sihyeon Kim<sup>1</sup>, Yunyang Xiong<sup>2</sup>, Hyunwoo J. Kim<sup>1†</sup>

<sup>1</sup>Korea University, <sup>2</sup>Meta Reality Labs

{lpmn678, cat0626, sh\_bs15, hyunwoojkim}@korea.ac.kr

yunyang@fb.com

## Abstract

Transformers have shown superior performance on various computer vision tasks with their capabilities to capture long-range dependencies. Despite the success, it is challenging to directly apply Transformers on point clouds due to their quadratic cost in the number of points. In this paper, we present a **Self-Positioning point-based Transformer (SPoTr)**, which is designed to capture both local and global shape contexts with reduced complexity. Specifically, this architecture consists of local self-attention and self-positioning point-based global cross-attention. The self-positioning points, adaptively located based on the input shape, consider both spatial and semantic information with disentangled attention to improve expressive power. With the self-positioning points, we propose a novel global cross-attention mechanism for point clouds, which improves the scalability of global self-attention by allowing the attention module to compute attention weights with only a small set of self-positioning points. Experiments show the effectiveness of SPoTr on three point cloud tasks such as shape classification, part segmentation, and scene segmentation. In particular, our proposed model achieves an accuracy gain of 2.6% over the previous best models on shape classification with ScanObjectNN. We also provide qualitative analyses to demonstrate the interpretability of self-positioning points. The code of SPoTr is available at <https://github.com/mlvlab/SPoTr>.

## 1. Introduction

Point clouds have been widely applied in various areas such as autonomous driving, robotics, and augmented reality. Since the point cloud is an unordered set of points with irregular structures, adopting convolutional neural networks (CNNs) on point clouds is challenging. Some works devoted effort to transforming point clouds into regular

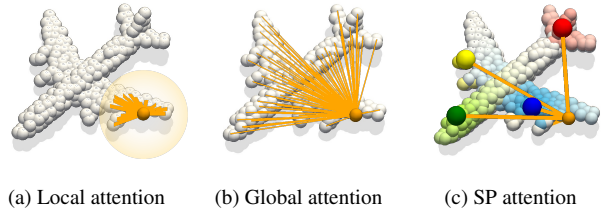


Figure 1. **Comparison of attention methods.** (a) Local attention, (b) Global attention, (c) Self-positioning point-based attention (SP attention).

structures, such as projection to multi-view images [1, 2] and voxelization [3, 4]. Others have tried to preserve the structure and design a convolution on the point space [5–11]. However, the ability to capture long-range dependencies is limited in most convolution-based approaches, while it is crucial to understand global shape context, especially with real-world data [12].

Transformer [13] tackled the long-range dependency issue in natural language processing and later it has been actively extended to 2D image processing [14–17]. Early works tried to replace convolutional layers with self-attention [14, 18–22], but they struggled with the quadratic computational cost of self-attention to the number of pixels. To mitigate the scalability issue, self-attention in local neighborhoods [15, 17] or approximating a self-attention with a reduced set of queries or keys [16, 23, 24] have been studied. For point clouds, Point Transformer [25] applies a local attention operation (Figure 1a) and PointASNL [26] employs a global attention module in a non-local manner (Figure 1b). Still, in point clouds, Transformer, which tackles both long-range dependency and scalability issues, has been less explored.

In this paper, we propose **Self-Positioning point-based Transformer (SPoTr)** to capture both local and global shape contexts with reduced complexity. SPoTr block consists of two attention modules: (i) *local points attention* (LPA) to learn local structures and (ii) *self-positioning point-based*

\*First two authors have equal contribution.

†is the corresponding author.

*attention* (SPA) to embrace global information via self-positioning points. SPA performs global attention by computing attention weights with only a small set of Self-Positioning points (SP points) instead of the whole input points different from the standard global attention as illustrated in Figure 1c. Specifically, SP points are adaptively located based on the input shape to cover the overall shape with only a small set of points. SP points learn its representation considering both spatial and semantic proximity through *disentangled attention*. Then, SPA non-locally distributes information of SP points to each input point. We also show that our SPoTr block generalizes set abstraction [27] with improved expressive power.

Further, we propose SPoTr architecture for standard point cloud tasks (*e.g.*, shape classification and semantic segmentation). We conduct extensive experiments with three datasets: ScanObjectNN [12], SN-Part [28], and S3DIS [29]. Our proposed method shows its effectiveness on all datasets compared to other attention-based methods. In particular, our architecture achieves an accuracy improvement of 2.6% over the previous best model in shape classification with a real-world dataset ScanObjectNN. Additionally, we demonstrate the effectiveness and interpretability of self-positioning point-based attention with qualitative analyses.

The **contribution** of our paper can be summarized as the following:

- We design a novel Transformer architecture (SPoTr) to tackle the long-range dependency issues and the scalability issue of Transformer for point clouds.
- We propose a global cross-attention mechanism with flexible self-positioning points (SPA). SPA aggregates information on a few self-positioning points via disentangled attention and non-locally distributes information to semantically related points.
- SPoTr achieves the best performance on three point cloud benchmark datasets (SONN, SN-Part, and S3DIS) against strong baselines.
- Our qualitative analyses show the effectiveness and interpretability of SPA.

## 2. Related works

**Deep learning on point clouds.** The success of CNNs has encouraged adapting CNNs to operate on point clouds rather than using hand-designed features. Early approaches aim to transform the unstructured point cloud data into a structured form for directly applying convolution. These include [1, 30–34], where they project 3D point clouds to 2D multi-view images for applying 2D convolution. Other approaches [3, 4, 35] convert point clouds to voxel grids,

then apply 3D convolution. However, both approaches have difficulty in preserving intrinsic geometries of point clouds. To address this issue, PointNet [36] directly processes point clouds with multi-layer perceptrons and a max-pooling function. However, it blindly aggregates all points without considering local information. Thus, PointNet++ [27] proposes utilizing local information through set abstraction and local grouping. For further understanding of local contexts, recent works [6–11, 37–40] have proposed explicit convolution kernels on the point space. KPConv [6] has applied deformable convolution [41, 42] to capture local information of point clouds. PointNeXt [43] has revisited PointNet++ by fully exploring its potential with improved training and augmentation schemes. Although the representation power has been improved by capturing local information, the ability to capture long-range dependencies is limited. SPoTr is the Transformer for point clouds equipped with global cross-attention to capture long-range dependencies.

**Attention-based methods on 2D images.** Following the success of self-attention and Transformers [13] in natural language understanding, many efforts have been made in the computer vision to replace convolution layers with self-attention layers [14, 18–22]. Despite the success, self-attention requires the quadratic computational cost with respect to the input image size. To address the scalability issue, several works adopt self-attention within local neighborhoods [15, 17]. Swin Transformer [15] utilizes non-overlapping windows and performs self-attention within each local window to get linear computational complexity in the number of input pixels. Other works explore global attention mechanisms with a small set of queries or keys to reduce complexity [16, 23, 24]. Twins [16] applies attention with a small set of representatives. Inspired by recent works, we suggest an efficient global cross-attention with only a small set of self-positioning points for point clouds.

**Attention-based methods on point clouds.** Recently, [25, 26, 44–50] have adopted attention operations for point cloud processing. PointASNL [26] leverages the attention operation to non-locally influence entire points. RPNNet [47] proposes attention-based modules for capturing local semantic and positional relations. PointTransformer [25] performs self-attention only within local neighborhoods. CloudTransformer [48] inspired by spatial transformer [51], uses an attention mechanism to transform the point cloud into a voxel grid for convolutional operation. Although these works have proven to be effective, most works have neglected the capability of Transformers to capture long-range dependencies due to their quadratic computational cost to the number of input points. In this paper, we aim to design Transformer architecture to capture both local and global information with a modest computational cost.

### 3. Method

The goal of our framework SPoTr is to learn point representations for various point cloud processing tasks with a Transformer architecture using *self-positioning points*. First, we shortly describe the background regarding the point-based approaches including PointNet++ and Point Transformer (Section 3.1). Second, we propose self-positioning point-based attention to efficiently capture the global context (Section 3.2). Third, we delineate the SPoTr block, which compromises both global cross-attention and local self-attention, and discuss the relation with a popular point-based network (Section 3.3). Finally, we present the overall architecture of SPoTr, which is composed of multiple SPoTr blocks, for shape classification and segmentation tasks (Section 3.4).

#### 3.1. Backgrounds

In this subsection, we briefly revisit the point-based approaches such as PointNet++ [27] and Point Transformer [25].

**PointNet++** [27] captures local shape information through set abstraction and local grouping. Given that a point set  $\mathcal{P} = \{x_i\}_{i=1}^N$ , where  $x_i$  is the position of the  $i$ -th point, and its corresponding feature  $\mathbf{f}_i$ , PointNet++ proposed local set abstraction as follows:

$$\mathbf{f}'_i = \mathcal{A}(\{\mathcal{M}([\mathbf{f}_j; \phi_{ij}]), \forall j \in \mathcal{G}_i\}), \quad (1)$$

where  $\mathcal{M}$  is the mapping function (e.g., MLP),  $\mathcal{A}$  is the aggregation function such as max-pooling, and  $\mathcal{G}_i$  is the index set of the local group centered on the  $i$ -th point.

**Point Transformer** [25] leverages self-attention operations [52] to represent local point groups. Similar to PointNet++, Point Transformer leverages local grouping to represent local point groups with a self-attention mechanism as follows:

$$\begin{aligned} \mathbf{f}'_i &= \sum_{j \in \mathcal{G}_i} \mathbf{A}_{ij} \odot (\mathbf{W}_1 \mathbf{f}_j + \Delta_{ij}), \\ \mathbf{A}_{ij} &= \text{SoftMax}(\mathcal{M}(\mathbf{W}_2 \mathbf{f}_i - \mathbf{W}_3 \mathbf{f}_j + \Delta_{ij})), \end{aligned} \quad (2)$$

where  $\odot$  denotes element-wise multiplication,  $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$  are learnable weight matrices,  $\mathcal{M}$  is a mapping function such as multi-layer perceptron, and  $\Delta$  is a positional encoding. Point Transformer [25] has shown the advantage of the attention mechanism on point clouds only with ‘local attention’ since computing the global attention on whole input points is almost infeasible on large-scale data.

#### 3.2. Self-positioning point-based attention

We propose an efficient global attention called *self-positioning point-based attention* (SPA) to resolve the inherent limitation of Transformer for point clouds (e.g., scalability) as illustrated in Figure 2. Herein, SPA computes attention weights with only a small set of self-positioning points named *SP points*. Since the overall shape context is captured by only a small number of SP points, the position of the SP points  $\delta_s$  needs to be *flexible*, so that the points become sufficiently representative of local part shapes. With our method, the SP points are adaptively located according to the input shape (see Section 4.3 for visualizations). Similar to the offsets in deformable convolutional neural networks [41] calculated with the feature of each pixel, we calculate  $\delta_s$  with its latent vector  $\mathbf{z}_s$  and feature  $\mathbf{f}_i$  of all input points  $\forall x_i \in \mathcal{P}$ , where  $\mathcal{P}$  is the set of points. We adopt adaptive interpolation and compute the positions of SP points as

$$\delta_s = \sum_i \text{SoftMax}(\mathbf{f}_i^\top \mathbf{z}_s) x_i. \quad (3)$$

Hence, SP points are always located nearby input points and precisely they stay within the convex hull of input points.

Then, SPA, equipped with SP points, performs global cross-attention in two steps: aggregation and distribution. At the aggregation step, SPA aggregates the features from all input points considering both spatial and semantic proximity. It can be written as

$$\psi_s = \sum_i g(\delta_s, x_i) \cdot h(\mathbf{z}_s, \mathbf{f}_i) \cdot \mathbf{f}_i, \quad (4)$$

where  $g, h$  are spatial and semantic kernel functions, respectively. For the spatial kernel function  $g$ , we use the Radial Basis Function (RBF) as

$$g(\delta_s, x_i) = \sum_i \exp(-\gamma \|\delta_s - x_i\|^2), \quad (5)$$

where  $\gamma \in \mathbb{R}_{++}$  is a bandwidth that adjusts the size of receptive fields. If  $\gamma$  has a higher value, the size of receptive fields gets smaller. For the semantic kernel function  $h$ , we utilize the attention-based kernel as below:

$$h(\mathbf{z}_s, \mathbf{f}_i) = \frac{\exp(\mathbf{f}_i^\top \mathbf{z}_s)}{\sum_{i'} \exp(\mathbf{f}_{i'}^\top \mathbf{z}_s)}. \quad (6)$$

Only considering the spatial information can cause *information smoothing*, i.e., the information from neighboring points with different semantics can reduce the descriptive power. Thus, we consider both spatial proximity and semantic proximity through two separate kernels  $h$  and  $g$ . These separate kernels can be interpreted as *disentangled attention*. The disentangled attention, which works similarly to the bilateral filter, allows the SP points to have

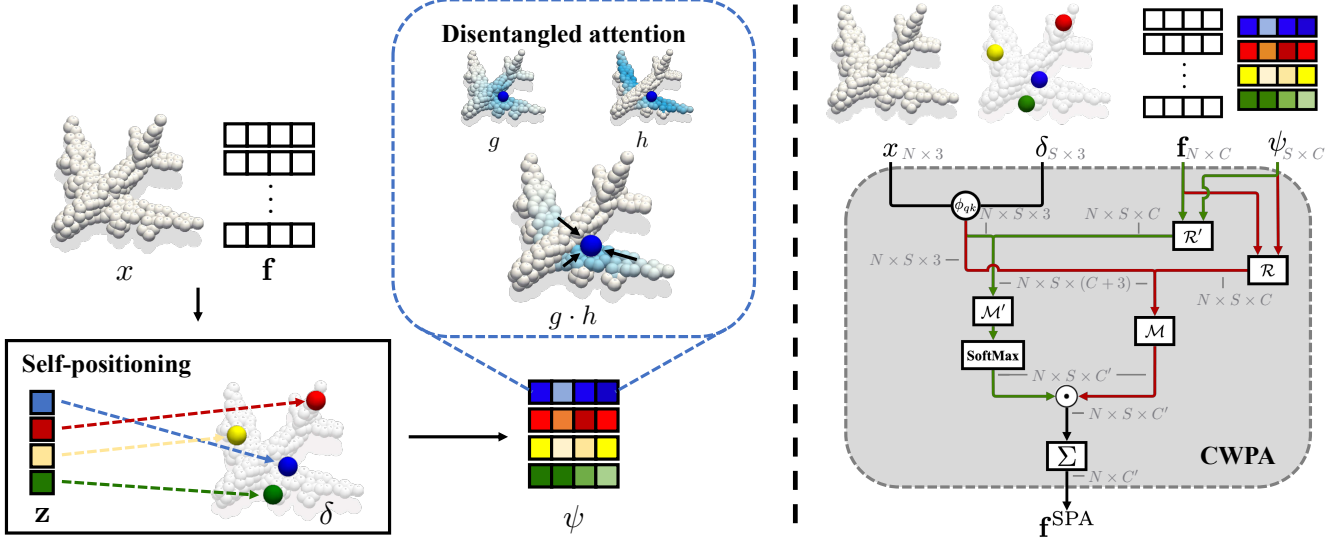


Figure 2. **Illustration of self-positioning point-based attention (SPA).** Given input points  $x$  and their corresponding features  $\mathbf{f}$ , self-positioning points (SP points)  $\delta$  are adaptively placed through the learnable latent  $\mathbf{z}$ . SP points aggregate features considering both spatial and semantic proximity and constructs  $\psi$  via disentangled attention. Then, SPA performs channel-wise point attention (CWPA) between input points and SP points to generate the output features  $\mathbf{f}^{\text{SPA}}$ .

greater descriptive power. Further analyses on how it processes, can be seen in Section 4.3. Finally, at the distribution step, SPA performs the cross-attention between SP points and input points as follows:

$$\mathbf{f}_i^{\text{SPA}} = \text{CWPA} \left( x_i, \mathbf{f}_i, \{\delta_s\}_{s=1}^S, \{\psi_s\}_{s=1}^S \right), \quad (7)$$

where  $\mathbf{f}_i^{\text{SPA}}$  is the final output of the SPA, CWPA is channel-wise point attention. CWPA will be described next.

**Channel-wise Point Attention.** Channel-Wise Point Attention (CWPA) computes attention weight between query and key points for each channel, different from the standard attention that generates the same attention weight across channels. CWPA is formulated as follows:

$$\begin{aligned} & \text{CWPA} \left( x_q, \mathbf{f}_q, \{x_k\}_{k \in \Omega_{\text{key}}}, \{\mathbf{f}_k\}_{k \in \Omega_{\text{key}}} \right) \\ &= \sum_{k \in \Omega_{\text{key}}} \mathbb{A}_{q,k,:} \odot \mathcal{M}([\mathcal{R}(\mathbf{f}_q, \mathbf{f}_k); \phi_{qk}]), \end{aligned} \quad (8)$$

where  $x_q, x_k \in \mathbb{R}^3$  are the positions of the query/key points and  $\mathbf{f}_q, \mathbf{f}_k \in \mathbb{R}^C$  are their corresponding features.  $\Omega_{\text{key}}$  denotes the set of key points. To take into account the relative information of contexts and positions, we use  $\phi_{qk}$  and  $\mathcal{R}(\cdot)$ .  $\phi_{qk}$  is a normalized relative position of the key point based on the query point,  $\mathcal{R}$  is the relation function between query and key feature (e.g.,  $\mathbf{f}_q - \mathbf{f}_k$ ) and  $\mathcal{M}$  indicates the mapping function. The channel-wise point attention  $\mathbb{A}_{q,k,:} \in \mathbb{R}^C$  be-

tween the query and key point is defined as follows:

$$\mathbb{A}_{q,k,c} = \frac{\exp(\mathcal{M}'([\mathcal{R}'(\mathbf{f}_q, \mathbf{f}_k); \phi_{qk}]/\tau)_c)}{\sum_{k' \in \Omega_{\text{key}}} \exp(\mathcal{M}'([\mathcal{R}'(\mathbf{f}_q, \mathbf{f}_{k'}); \phi_{qk'}]/\tau)_c)}, \quad (9)$$

where  $c$  is the index of channel, and  $\tau$  denotes temperature.  $\mathcal{M}'$  and  $\mathcal{R}'$  are the mapping function and the relation function, respectively. In our implementation, we use  $\mathcal{R}' = \mathbf{f}_q - \mathbf{f}_k$  same as  $\mathcal{R}$ . By adopting the proposed channel-wise point attention, CWPA can learn more powerful and flexible representations compared to standard attention.

### 3.3. Self-positioning point-based Transformer

We now propose the SPoTr block that utilizes *self-positioning point-based attention (SPA)* with *local point attention (LPA)*. By combining LPA and SPA, it captures not only local and short-distance information but also long-distance and global information.

**Local points attention (LPA).** We adopt local points attention (LPA) defined on a local group to learn local shape context. A local point group consists of neighbors in ball query centered on an anchor point  $x_i$ . The attention for each local point group  $\mathcal{G}_i$  and points  $\{x_j | j \in \mathcal{G}_i\}$  is defined as

$$\mathbf{f}_i^{\text{LPA}} = \text{CWPA} \left( x_i, \mathbf{f}_i, \{x_j\}_{j \in \mathcal{G}_i}, \{\mathbf{f}_j\}_{j \in \mathcal{G}_i} \right), \quad (10)$$

where  $\mathbf{f}_j$  is the feature vector of point  $x_j$ ,  $\mathbf{f}_i^{\text{LPA}}$  is an output feature vector of LPA. We adopt channel-wise point attention operation (CWPA) same as SPA.



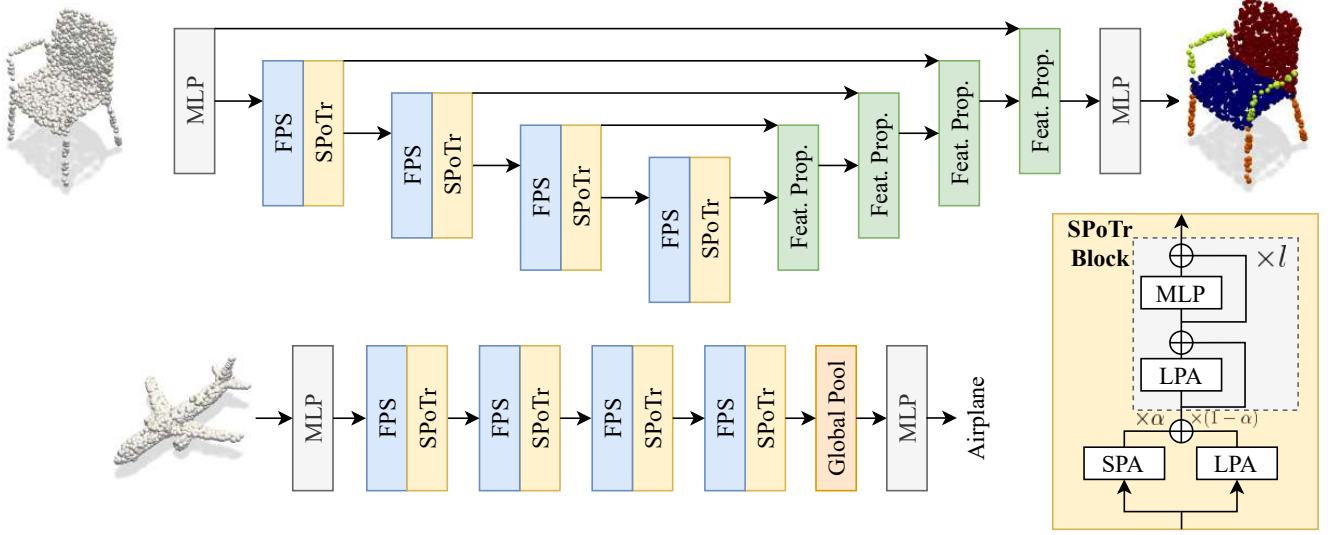


Figure 3. **Overall Architecture of SPoTr.** For classification (bottom), four SPoTr blocks run consecutively, followed by a max-pooling and a multi-layer perceptron. For segmentation (top), a U-net style architecture is adopted with SPoTr blocks for downsampling and feature propagation for upsampling, followed by a multi-layer perceptron.

**SPoTr block.** We construct a SPoTr block by combining the local points attention (LPA) module and self-positioning point-based attention (SPA) module to capture local and global information simultaneously. As shown in Figure 3 (bottom right), the SPoTr block is defined as follows:

$$\hat{\mathbf{f}}_i = \alpha \cdot \mathbf{f}_i^{\text{SPA}} + (1 - \alpha) \cdot \mathbf{f}_i^{\text{LPA}} \quad (11)$$

where  $\alpha$  is a learnable parameter that softly selects the representations generated by self-positioning point-based attention and local points attention. Finally, LPA and MLP with batch normalization and the residual connection are applied to extract point-wise high-level representations.

**Connection between SPoTr block and Set abstraction.** We show the superiority of the SPoTr block by discussing it with set abstraction in PointNet++ [27].

**Remark 1** A SPoTr block with proper  $\alpha, \mathcal{M}, \mathcal{M}', \mathcal{R}, \mathcal{R}'$  can express set abstraction.

When the value of  $\tau$  is sufficiently low, the function  $\mathcal{M}'$  is the same as  $\mathcal{M}$ ,  $\alpha = 0$ , and  $\mathcal{R}(\mathbf{f}_q, \mathbf{f}_k) = \mathcal{R}'(\mathbf{f}_q, \mathbf{f}_k) = \mathbf{f}_k$ , the channel-wise point attention becomes equivalent to the set abstraction. In this setting, the attention weight between the query point and  $k$ -th key point on  $c$ -th channel becomes 1 if  $k = \arg\max_{k' \in \Omega_k} \mathcal{M}([\mathbf{f}_{k'}; \phi_{qk'}])_{q,k',c}$ . Otherwise, the attention score is 0. It means that the attention only activates the maximum channels alike a max-pooling operation. Therefore, the SPoTr block can play a role as a max-pooling operation following the mapping function, which is the set ab-

straction. This fact supports the improved expressive power of SPoTr on point cloud analysis.

### 3.4. SPoTr architectures

We design a Transformer-based architecture called SPoTr for point cloud tasks as illustrated in Figure 3.

**Classification.** For the shape classification task, we build our Transformer encoder by stacking the SPoTr blocks described in Section 3.3. To increase the representational power, we first apply an MLP before operating the attention blocks following [25]. Then, the SPoTr blocks are sequentially applied on sampled points, which are sampled through the farthest point sampling (FPS). In shape classification, we set  $l = 0$  for the SPoTr block since we empirically found that it is enough in shape classification task. Besides, the sampling rates are 1/4 for every stage. We use the ball query that selects points within a radius (an upper limit of the number of neighborhoods is set in implementation) to generate a local point group  $\mathcal{G}_i$  centered at point  $x_i$  following [27]. After the last stage, the features are aggregated by a max-pooling function and processed by an MLP.

**Segmentation.** The encoder for semantic segmentation contains the SPoTr blocks and FPS. Following previous studies [27], we apply a U-net designed architecture, which contains the feature propagation layers and the SPoTr blocks for dense prediction. Same to the classification, we use a ball query to generate a local group. The outputs of the final block are processed by an MLP. More details on SPoTr architectures are in the supplement.

Methods	Year	mAcc	OA
PointNet [36]	2017	63.4	68.2
PointNet++ [27]	2017	75.4	77.9
SpiderCNN [9]	2018	69.8	73.7
PointCNN [7]	2018	75.1	78.5
DGCNN [53]	2019	73.6	78.1
DRNet [54]	2021	78.0	80.3
GBNet [55]	2021	77.8	80.5
SimpleView [33]	2021	-	80.5
PRA-Net [56]	2021	77.9	81.0
MVTN [34]	2021	-	82.8
CT [48]	2021	83.1	85.5
PointMLP [39]	2022	84.4	85.7
RepSurf-U [40]	2022	83.1	86.0
PointNeXt [43]	2022	85.8±0.6	87.7±0.4
<b>SPoTr</b>	2023	<b>86.8</b>	<b>88.6</b>

Table 1. **Shape classification results on PB\_T50\_RS in SONN.** mAcc is the mean of class accuracy and OA is the overall accuracy.

## 4. Experiments

In this section, we demonstrate the effectiveness of SPoTr and provide quantitative and qualitative results for further analyses. First, we conduct shape classification and semantic segmentation (Section 4.1). We also provide ablation studies and complexity analysis of SPoTr (Section 4.2). Lastly, we provide visualizations to better understand how SPA behaves (Section 4.3). Implementation details are available in the supplement.

### 4.1. Shape classification and semantic segmentation

**Shape Classification.** For the shape classification, we validate SPoTr on a real-world dataset ScanObjectNN (SONN) [12]. SONN has 2,902 objects categorized into 15 classes from SceneNN [57] and ScanNet [58]. Among diverse variants of SONN, we use PB\_T50\_RS (SONN\_PB), which is the most challenging version with random perturbation and contains 14,510 objects in total. We follow the official split of [12], where they divide SONN into 80% for training and 20% for evaluation. Also, we sample 1,024 points for training and evaluating the models.

Table 1 shows that SPoTr outperforms all baselines with the mean of class accuracy (mAcc) of 86.8% and overall accuracy (OA) of 88.6% (+1.0% mAcc, +0.9% OA). This result shows that capturing long-range context is important for recognizing 3D shapes in real-world datasets.

**Part Segmentation.** For part segmentation, we use SN-Part [28, 60], which is a synthetic dataset with 16,881 shapes from 16 categories with 50 part labels. We follow the split used in [36], where 14,006 samples are for training

Methods	Year	cls. mIoU	ins. mIoU
PointNet [36]	2017	80.4	83.7
PointNet++ [27]	2017	81.9	85.1
PointCNN [7]	2018	84.6	86.1
DGCNN [53]	2019	82.3	85.1
RSCNN [5]	2019	84.0	86.2
KPConv [6]	2019	85.1	86.4
PointConv [10]	2019	82.8	85.7
PointASNL [26]	2020	-	86.1
PCT [46]	2021	-	86.4
PAConv [11]	2021	84.6	86.1
AdaptConv [38]	2021	83.4	86.4
PointTransformer [25]	2021	83.7	86.6
CurveNet [50]	2021	-	86.8
PointMLP [39]	2022	84.6	86.1
PointNeXt [43]	2022	85.2 ± 0.1	87.0 ± 0.1
<b>SPoTr</b>	2023	<b>85.4</b>	<b>87.2</b>

Table 2. **Part segmentation results on SN-Part.** ins. mIoU is the mean of instance IoU. cls. mIoU is the mean of the class IoU.

Methods	Year	OA	mAcc	mIoU
PointNet [36]	2017	-	-	41.1
PointCNN [7]	2018	85.9	63.9	57.3
PointWeb [59]	2019	87.0	66.6	60.3
KPConv [6]	2019	-	72.8	67.1
PCT [46]	2021	-	67.7	61.3
CT [48]	2021	-	-	67.9
PointTransformer [25]	2021	<b>90.8</b>	-	70.4
RepSurf-U [40]	2022	90.2	76.0	68.9
PointNeXt [43]	2022	90.6 ± 0.1	-	70.5 ± 0.3
<b>SPoTr</b>	2023	90.7	<b>76.4</b>	<b>70.8</b>

Table 3. **Semantic segmentation results on S3DIS.** OA is the overall accuracy, mAcc is the mean of class accuracy, and mIoU is the mean of instance IoU.

and 2,874 samples are for validation. On each shape, 2,048 points are randomly sampled.

The results are reported in Table 2, where we evaluate the performance with the mean of instance IoU (ins. mIoU) and class IoU (cls. mIoU). Following previous works [5, 11, 50], we report the results with a multi-scale inference setting. Although the performance in SN-Part is quite saturated, SPoTr achieves the best performance 87.2% with considerable improvements (+0.2% mIoU).

**Scene Segmentation.** For comparison with previous methods [6, 25, 36, 49] on scene segmentation, we validate SPoTr on the widely used benchmark dataset S3DIS [29]. S3DIS is the large-scale dataset containing 271 rooms from

Method	$g$	$h$	SP	OA
w/o SPA ( <i>baseline</i> )				87.9
w/o self-positioning	✓	✓		87.7
w/o disentangled attention	✓		✓	88.2
SPoTr ( <i>ours</i> )	✓	✓	✓	<b>88.6</b>

Table 4. **Ablations on SONN-PB.**  $g$ : spatial kernel,  $h$ : semantic kernel, SP: self-positioning points. OA is the overall accuracy.

Attention type	Semantic rel. $\mathcal{R}$	OA
Standard Att.	—	86.1
CWPA	$\mathbf{f}_k$	88.1
CWPA	$\mathbf{f}_q + \mathbf{f}_k$	86.4
CWPA	$\mathbf{f}_q \odot \mathbf{f}_k$	85.4
CWPA	$\mathbf{f}_q - \mathbf{f}_k$	<b>88.6</b>

Table 5. **Performance comparisons of different attention types and semantic relation  $\mathcal{R}$  on SONN-PB.** Attention types : Standard Attention in Transformer [13] and channel-wise point attention (CWPA) with Semantic relation :  $\mathcal{R}(\mathbf{f}_q, \mathbf{f}_k)$

6 indoor areas with 13 semantic categories. In our experiments, we largely follow the settings of PointTransformer [25] and consider Area-5 as the test set.

As shown in Table 3, SPoTr outperforms all previous methods in every metric (*i.e.*, overall accuracy (OA), mean of class accuracy (mAcc), and mean of instance IoU (mIoU)). The superior performance over previous Transformer architecture [25] (+0.4% mIoU) proves the importance of long-range dependency in the semantic segmentation as well as the shape classification.

## 4.2. Quantitative analysis

**Ablation studies.** We explore how self-positioning positions (SP) and disentangled attention contribute to SPA. Table 4 shows the final results on SONN, where the baseline (*w/o SPA*) learns only with local point attention. In the case of *w/o self-positioning*, we use FPS to randomly select a small set of points for cross-attention, and for *w/o disentangled attention*, we only adopt the spatial kernel function  $g$ . Our model with all the components of SPA achieves the best performance of 88.6% in overall accuracy. This superior performance verifies that every component is crucial for SPA. In particular, when we use FPS instead of SP, the performance is even worse than the baseline as overall accuracy dropped from 87.9% to 87.7%. This observation suggests the positions of SP points *matter* for global cross-attention. Rather than simple sampling, our learnable approach successfully locates SP points and makes global cross-attention effective. Next, with *w/o disentangled attention*, the performance gain in OA is minimal (0.3%) over the

Method	Param ↓ (M)	FLOPs ↓ (G)	Memory ↓ (GB)	Throughput ↑ (shapes/s)
GSA	<b>1.7</b>	114.0	24.2	17.7
SPA ( <i>ours</i> )	<b>1.7</b>	<b>10.8</b>	<b>2.5</b>	<b>281.5</b>
	(-)	(- 90.5%)	(- 89.7%)	( $\times 15.9$ )

Table 6. **Complexity analysis on SN-Part.** SPA: self-positioning point-based attention, GSA: global self-attention.

baseline compared to using disentangled attention (0.7%). It indicates that disentangled attention improves the descriptive power by filtering semantically irrelevant information.

**Attention types and semantic relation  $\mathcal{R}$ .** In Table 5, we conduct experiments to compare the models with different attention types (Standard attention in Transformer [13] and our CWPA) and semantic relations ( $\mathcal{R}(\mathbf{f}_q, \mathbf{f}_k) = \mathbf{f}_k$ ,  $\mathbf{f}_q + \mathbf{f}_k$ ,  $\mathbf{f}_q \odot \mathbf{f}_k$ , and  $\mathbf{f}_q - \mathbf{f}_k$ ). The models adopting the CWPA outperform the model with the standard attention, which shows that the channel-wise point attention operation is more powerful to represent point clouds compared to the standard attention. Furthermore, the results demonstrate that Sub ( $\mathbf{f}_q - \mathbf{f}_k$ ) is most appropriate to model the semantic relation between points.

**Complexity analysis on SN-Part.** We analyze the space and time complexity to validate the computational efficiency of SPoTr during inference time with a batch size of 8. For a baseline, SPA in SPoTr is replaced by the standard global self-attention (abbreviated in GSA) with CWPA. For a comparison with GSA requiring the quadratic complexity, we inevitably use the variants of SPoTr, where the channel size of each layer is reduced by  $\times 1/4$ . For space complexity, we measure the number of parameters and total memory usage, and for time complexity, we measure FLOPs and throughput performance. Table 6 empirically proves the efficiency over GSA. For space complexity, GSA shares a similar number of parameters with SPA but introduces a large memory usage of 24.2 (GB). Instead, Our SPA only uses 2.5 (GB) (-89.7%). Also, SPA largely reduces the time complexity from 114.0 GFLOPS with a throughput of 17.7 (shapes/s) to 10.8 GFLOPS (-90.5%) with a throughput of 281.5 (shapes/s) ( $\times 15.9$ ).

## 4.3. Qualitative analysis

For a deeper understanding of each component in SPA, such as self-positioning points (SP points) and disentangled attention, we provide qualitative results in this section. We use SN-Part for visualizations.

**Self-positioning points.** As mentioned in Section 3.2, it is important that SP points are adaptively located considering

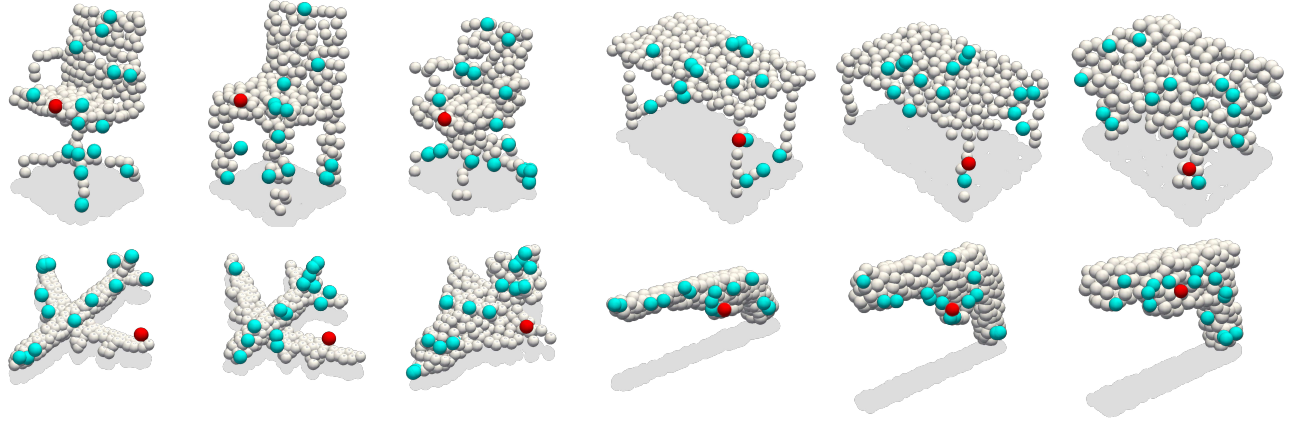


Figure 4. **Self-positioning points (SP points).** **SP points** are adaptively self-positioned according to each shape. **Red points** correspond to specific SP points. Under the same class, the red points are located at *semantically similar* positions.

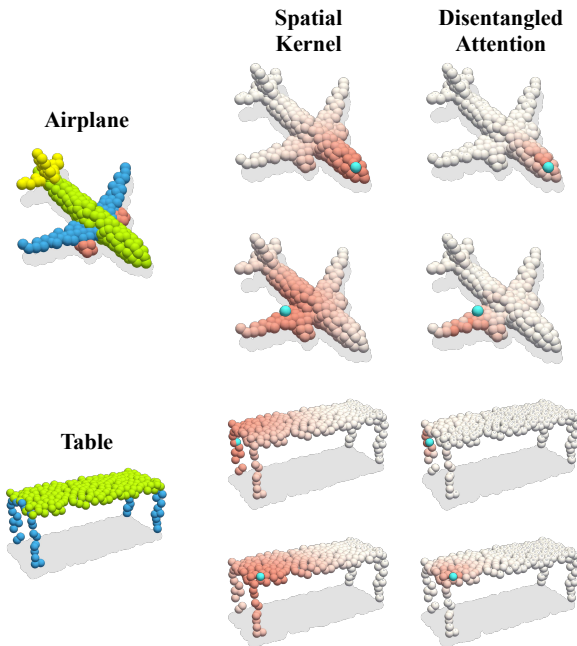


Figure 5. **Visual comparison of disentangled attention with a spatial kernel.** A spatial kernel (Middle) only considers spatial proximity without considering semantic relevance. Differently, Disentangled Attention (Right) filters out irrelevant information.

the input shape. Figure 4 shows that the SP points are adaptively located on various samples from different categories. Interestingly, a specific SP point (colored in red) appears at a *semantically similar* place for each category. For example, red dots from airplanes are always near the left-wing. This consistent placement of SP points implies that each SP point learns to represent semantically similar regions.

**Disentangled attention.** SPA aggregates feature considering spatial proximity as well as semantic proximity via disentangled attention as introduced in Section 3.2. Figure 5 shows the weights of the spatial kernel  $g$  and the effective receptive field of the disentangled attention  $g \cdot h$ . Cyan-colored points are selected SP points and kernel weights are illustrated in heatmaps. With only the spatial kernel  $g$ , SP point blindly aggregates the information of neighbor points inducing *irrelevant* information from close regions (e.g., a wing and a body of an airplane are strongly colored in the second row of the figure). Conversely, with our disentangled attention  $g \cdot h$ , the same SP point selectively aggregates information considering both spatial proximity and semantic proximity (e.g., the right-wing is only colored in the figure). The obvious difference suggests disentangled attention is crucial for enhancing the descriptive power by suppressing irrelevant information.

## 5. Conclusion

In this paper, we propose SPoTr, a Transformer for point clouds, which captures both local and global shape context without the quadratic complexity of input points. SPoTr includes two attention modules: self-positioning point-based attention (SPA) and local points attention (LPA). SPA is a novel global cross-attention, which aggregates information via disentangled attention and non-locally distributes information to entire points. Our experiments show superior performance across various tasks, including ScanObjectNN, SN-Part, and S3DIS.

**Acknowledgements.** This work was supported by the MSIT, Korea, under the ICT Creative Consilience program (IITP-2023-2020-0-01819) supervised by the IITP and the Virtual Engineering Platform Project (P0022336) of the Ministry of Trade, Industry and Energy (MOTIE), Korea.



## References

- [1] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015. 1, 2
- [2] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 1
- [3] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, 2015. 1, 2
- [4] Yin Zhou and Oncel Tuzel. Voxnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 1, 2
- [5] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, 2019. 1, 6
- [6] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotequi, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 1, 2, 6
- [7] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *NeurIPS*, 2018. 1, 2, 6
- [8] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *TOG*, 37(4):71:1–71:12, 2018. 1, 2
- [9] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *ECCV*, 2018. 1, 2, 6
- [10] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, 2019. 1, 2, 6
- [11] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In *CVPR*, 2021. 1, 2, 6
- [12] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *ICCV*, 2019. 1, 2, 6
- [13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2, 7
- [14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2
- [15] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1, 2
- [16] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. In *NeurIPS*, 2021. 1, 2
- [17] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021. 1, 2
- [18] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019. 1, 2
- [19] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *ICCV*, 2019. 1, 2
- [20] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018. 1, 2
- [21] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pre-training from pixels. In *ICML*, 2020. 1, 2
- [22] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *ICLR*, 2020. 1, 2
- [23] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. In *ICLR*, 2022. 1, 2
- [24] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 1, 2
- [25] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021. 1, 2, 3, 5, 6, 7
- [26] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *CVPR*, 2020. 1, 2, 6
- [27] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2, 3, 5, 6
- [28] 3D Warehouse. Sketchup. <https://3dwarehouse.sketchup.com/>, 2022. 2, 6
- [29] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *CVPR*, 2016. 2, 6
- [30] Haiyun Guo, Jinqiao Wang, Yue Gao, Jianqiang Li, and Hanqing Lu. Multi-view 3d object retrieval with deep embedding network. *TIP*, 25(12):5526–5537, 2016. 2
- [31] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, 2016. 2

- [32] Maximilian Jaritz, Jiayuan Gu, and Hao Su. Multi-view pointnet for 3d scene understanding. In *ICCVW*, 2019. 2
- [33] Ankit Goyal, Hei Law, Bowei Liu, Alejandro Newell, and Jia Deng. Revisiting point cloud shape classification with a simple and effective baseline. In *ICML*, 2021. 2, 6
- [34] Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. Mvtn: Multi-view transformation network for 3d shape recognition. In *ICCV*, 2021. 2, 6
- [35] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *NeurIPS*, 2019. 2
- [36] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2, 6
- [37] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *CVPR*, 2018. 2
- [38] Haoran Zhou, Yidan Feng, Mingsheng Fang, Mingqiang Wei, Jing Qin, and Tong Lu. Adaptive graph convolution for point cloud analysis. In *ICCV*, 2021. 2, 6
- [39] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Re-thinking network design and local geometry in point cloud: A simple residual MLP framework. In *ICLR*, 2022. 2, 6
- [40] Haoxi Ran, Jun Liu, and Chengjie Wang. Surface representation for point clouds. In *CVPR*, 2022. 2, 6
- [41] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 2, 3
- [42] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *CVPR*, 2019. 2
- [43] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In *NeurIPS*, 2022. 2, 6
- [44] Saining Xie, Sainan Liu, Zeyu Chen, and Zhuowen Tu. Attentional shapecontextnet for point cloud recognition. In *CVPR*, 2018. 2
- [45] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*, 2019. 2
- [46] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *CVM*, 7(2):187–199, 2021. 2, 6
- [47] Haoxi Ran, Wei Zhuo, Jun Liu, and Li Lu. Learning inner-group relations on point clouds. In *ICCV*, 2021. 2
- [48] Kirill Mazur and Victor Lempitsky. Cloud transformers: A universal approach to point cloud processing tasks. In *ICCV*, 2021. 2, 6
- [49] Jaesung Choe, Chunghyun Park, Francois Rameau, Jaesik Park, and In So Kweon. Pointmixer: Mlp-mixer for point cloud understanding. In *ECCV*, 2022. 2, 6
- [50] Tiange Xiang, Chaoyi Zhang, Yang Song, Jianhui Yu, and Weidong Cai. Walk in the cloud: Learning curves for point clouds shape analysis. In *ICCV*, 2021. 2, 6
- [51] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NeurIPS*, 2015. 2
- [52] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, 2020. 3
- [53] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 38(5):146:1–146:12, 2019. 6
- [54] Shi Qiu, Saeed Anwar, and Nick Barnes. Dense-resolution network for point cloud classification and segmentation. In *WACV*, 2021. 6
- [55] Shi Qiu, Saeed Anwar, and Nick Barnes. Geometric back-projection network for point cloud classification. *TMM*, 24:1943–1955, 2021. 6
- [56] Silin Cheng, Xiwu Chen, Xinwei He, Zhe Liu, and Xiang Bai. Pra-net: Point relation-aware network for 3d point cloud analysis. *TIP*, 30:4436–4448, 2021. 6
- [57] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. Scenenn: A scene meshes dataset with annotations. In *3DV*, 2016. 6
- [58] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 6
- [59] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *CVPR*, 2019. 6
- [60] Sanghyeok Lee, Minkyu Jeon, Injae Kim, Yunyang Xiong, and Hyunwoo J Kim. Sagemix: Saliency-guided mixup for point clouds. In *NeurIPS*, 2022. 6