# Relation-Shape Convolutional Neural Network for Point Cloud Analysis

Yongcheng Liu[†‡] Bin Fan[*†] Shiming Xiang[†‡] Chunhong Pan[†]
[†]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[‡]School of Artificial Intelligence, University of Chinese Academy of Sciences
Email: {yongcheng.liu, bfan, smxiang, chpan}@nlpr.ia.ac.cn

## Abstract

*Point cloud analysis is very challenging, as the shape implied in irregular points is difficult to capture. In this paper, we propose RS-CNN, namely, **R**elation-**S**hape **C**onvolutional **N**eural **N**etwork, which extends regular grid CNN to irregular configuration for point cloud analysis. The key to RS-CNN is learning from relation, i.e., the geometric topology constraint among points. Specifically, the convolutional weight for local point set is forced to learn a high-level relation expression from predefined geometric priors, between a sampled point from this point set and the others. In this way, an inductive local representation with explicit reasoning about the spatial layout of points can be obtained, which leads to much shape awareness and robustness. With this convolution as a basic operator, RS-CNN, a hierarchical architecture can be developed to achieve contextual shape-aware learning for point cloud analysis. Extensive experiments on challenging benchmarks across three tasks verify RS-CNN achieves the state of the arts.*

## 1. Introduction

Recently, the analysis of 3D point cloud has drawn a lot of attention, as it has many applications such as autonomous driving and robot manipulation. However, this task is very challenging, since it is difficult to infer the underlying shape formed by these irregular points (see Fig. 1 for detail).

For this issue, much effort is focused on replicating the remarkable success of convolutional neural network (CNN) on regular grid data (*e.g.*, image) analysis [17, 32], to irregular point cloud processing [26, 15, 45, 29, 27, 34, 38]. Some works transform point cloud to regular voxels [42, 22, 3] or multi-view images [35, 2, 5] for easy application of classic grid CNN. These transformations, however, usually lead to much loss of inherent geometric information in 3D point cloud, as well as high complexity.

To directly process point cloud, PointNet [24] independently learns on each point and gathers the final features for a global representation. Though impressive, this design ignores local structures that have been proven to be important for abstracting high-level visual concepts in image CNN [49]. To solve this problem, some works partition point cloud into several subsets by sampling [26] or superpoint [18]. Then a hierarchy is built to learn contextual representation from local to global. Nevertheless, this extremely relies on effective inductive learning of local subsets, which is quite intractable to achieve.

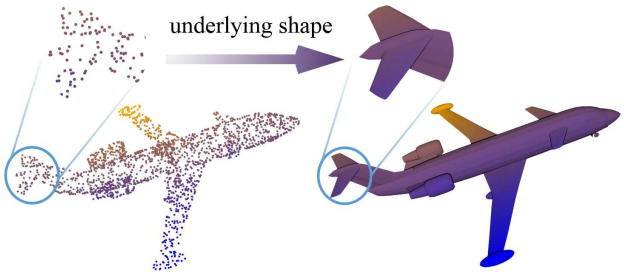


Figure 1. Left part: Point cloud. Right part: Underlying shape formed by this point cloud.

Generally, there are mainly three challenges for learning from point set $P \subset \mathbb{R}^3$: (1) $P$ is unordered, thus requiring the learned representation being permutation invariant; (2) $P$ distributes in 3D geometric space, thus demanding the learned representation being robust to rigid transformation (*e.g.*, rotation and translation); (3) $P$ forms an underlying shape, therefore, the learned representation should be of discriminative shape awareness. The issue (1) has been well resolved by symmetric function [24, 27, 48], while (2) and (3) still demand for a full exploration. The goal of this work is to extend regular grid CNN to irregular configuration for handling these issues together.

To this end, we propose a relation-shape convolutional neural network (aliased as RS-CNN). The key to RS-CNN is learning from relation, *i.e.*, the geometric topology constraint among points, which in our view can encode meaningful shape information in 3D point cloud.

Specifically, each local convolutional neighborhood is constructed by taking a sampled point $x$ as the centroid and

*Corresponding author: Bin Fan

the surrounding points as its neighbors $\mathcal{N}(x)$. Then, the convolutional weight is forced to learn a high-level relation expression from predefined geometric priors, *i.e.*, intuitive low-level relation between $x$ and $\mathcal{N}(x)$. By convoluting in this way, an inductive representation with explicit reasoning about the spatial layout of points can be obtained. It discriminatively reflects the underlying shape that irregular points form thus is shape-aware. Furthermore, it can benefit from geometric priors, including the invariance to points permutation and the robustness to rigid transformation (*e.g.*, translation and rotation). With this convolution as a basic operator, a hierarchical CNN-like architecture, *i.e.*, RS-CNN, can be developed to achieve contextual shape-aware learning for point cloud analysis.

The key contributions are highlighted as follows:

- A novel learn-from-relation convolution operator called relation-shape convolution is proposed. It can explicitly encode geometric relation of points, thus resulting in much shape awareness and robustness;
- A deep hierarchy equipped with the relation-shape convolution, *i.e.*, RS-CNN, is proposed. It can extend regular grid CNN to irregular configuration for achieving contextual shape-aware learning of point cloud;
- Extensive experiments on challenging benchmarks across three tasks, as well as thorough empirical and theoretical analysis, demonstrate RS-CNN achieves the state of the arts.

## 2. Related Work

**View-based and volumetric methods.** View-based methods represent a 3D shape as a group of 2D views from different angles. Recently, many works [35, 2, 5, 43, 6, 25] have been proposed to recognize these view images with deep neural networks. They often finetune a pre-trained image-based architecture for accurate recognition. However, 2D projections could cause loss of shape information due to self-occlusions, and it often demands a huge number of views for decent performance.

Volumetric methods convert the input 3D shape into a regular 3D grid, over which classic CNN can be employed [42, 22, 3]. The main limitation is the quantization loss of the shape due to the low resolution enforced by 3D grid. Recent space partition methods like K-d trees [16] or octrees [39, 36, 28] rescue some resolution issues but still rely on the subdivision of a bounding volume rather than a local geometric shape. In contrast to these methods, our work aims to process 3D point cloud directly.

**Deep learning on point cloud.** PointNet [24] pioneers this route by independently learning on each point and gathering the final features with max pooling. Yet this design neglects local structures, which have been proven important for the success of CNN. To remedy this, PointNet++ [26] suggests a hierarchical application of PointNet to multiple subsets of point cloud. Local structure exploitation with PointNet is also investigated in [4, 30]. In addition, Superpoint [18] is proposed to partition point cloud into geometric elements. Graph convolution network is applied on a local graph created by neighboring points [38, 37, 20]. However, these methods do not explicitly model the local spatial layout of points, thus acquiring less shape awareness. By contrast, our work captures the spatial layout of points by learning a high-level relation expression among points.

Some works map point cloud to a high-dimensional space to facilitate the application of classic CNN. SPLATNet [34] maps the input points onto a sparse lattice, then processing with bilateral convolution [14]. PCNN [1] extends the function over point cloud to a continuous volumetric function over ambient space. These methods could cause loss of geometric information, while our method directly operates on point cloud without introducing such loss.

Another key issue is the irregularity of points. Some works focus on analyzing symmetric functions that are equivariant to point sets learning [24, 27, 48, 19]. Some other works [24, 21] develop alignment network for the robustness to rigid transformation in 3D space. However, the alignment learning is a suboptimal solution for this issue. Some traditional descriptors like Fast Point Feature Histograms can be invariant to translation and rotation, yet they are often less effective for high-level shape understanding. Our method that learns on geometric relation among points is naturally robust to rigid transformation, whilst being highly effective due to the powerfulness of deep nets.

**Relation learning.** To learn a data-dependent weight from relation has been explored in the field of image and video analysis. Spatial transformer [13] learns a transition matrix to align 2D images. Non-local network [40] learns long-term relation across video frames. Relation networks [9] learn position relation across objects.

There are also some works focusing on the relation learning in 3D point cloud. DGCNN [41] captures similar local shapes by learning point relation in a high-dimensional feature space, yet this relation could be unreliable in some cases. Wang *et al.* propose a parametric continuous convolution that is based on computable relation among points, but they do not explicitly learn from local to global like classic CNN. By contrast, our method learns a high-level relation expression from geometric priors in 3D space, and performs contextual local-to-global shape learning.

## 3. Shape-Aware Representation Learning

The core of point cloud analysis is to discriminatively represent the underlying shape with robustness. Here we learn contextual shape-aware representation for this goal, by extending regular grid CNN to irregular configuration with a novel relation-shape convolution (RS-Conv).

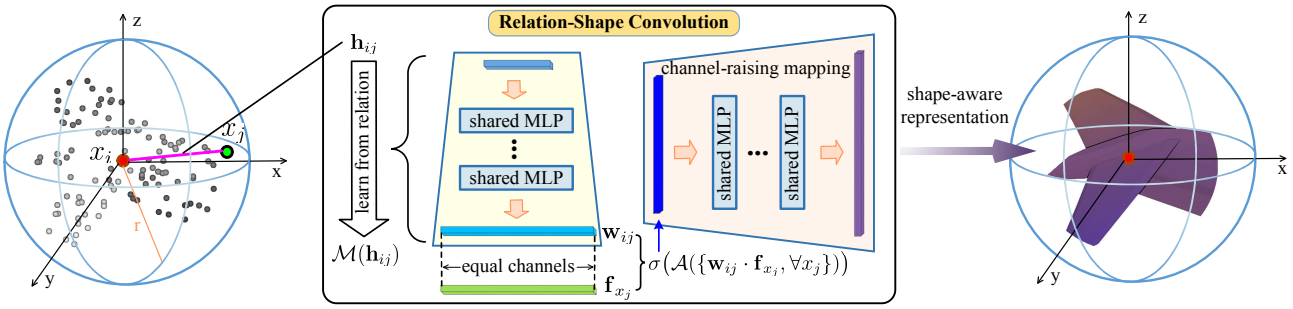


Figure 2. Overview of relation-shape convolution (RS-Conv). The key is to learn from relation. Specifically, the convolutional weight for $x_j$ is converted to $\mathbf{w}_{ij}$, which learns a mapping $\mathcal{M}$ (Eq. (2)) on predefined geometric relation vector $\mathbf{h}_{ij}$. In this way, the inductive convolutional representation $\sigma\big(\mathcal{A}(\{\mathbf{w}_{ij}\cdot\mathbf{f}_{x_j}, \forall x_j\})\big)$ (Eq. (3)) can expressively reason the spatial layout of points, resulting in discriminative shape awareness. As in image CNN [32], further channel-raising mapping is conducted for a more powerful shape-aware representation.

### 3.1. Relation-Shape Convolution

Local-to-global learning, which has gained remarkable success in image CNN [17, 32], is a promising solution for contextual shape representation. However, it extremely relies on shape-aware inductive learning from irregular point subsets, which remains a quite intractable problem.

**Modeling.** To overcome this issue, we model local point subset $P_{\text{sub}} \subset \mathbb{R}^3$ to be a spherical neighborhood, with a sampled point $x_i$ as the centroid and surrounding points as its neighbors $x_j \in \mathcal{N}(x_i)$. The left-most part of Fig. 2 illustrates this modeling. Then, our goal is to learn an inductive representation $\mathbf{f}_{P_{\text{sub}}}$ of this neighborhood, which should discriminatively encode the underlying shape information. To this end, we formulate a general convolutional operation as

$$\mathbf{f}_{P_{\text{sub}}} = \sigma\big(\mathcal{A}(\{\mathcal{T}(\mathbf{f}_{x_j}),\ \forall x_j\})\big)^{1},\ d_{ij} < r\ \forall x_j \in \mathcal{N}(x_i), \tag{1}$$

where $x$ is a 3D point and $\mathbf{f}$ is a feature vector. $d_{ij}$ is the Euclidean distance between $x_i$ and $x_j$, and $r$ is the sphere radius. Here $\mathbf{f}_{P_{\text{sub}}}$ is obtained by first transforming the features of all the points in $\mathcal{N}(x_i)$ with function $\mathcal{T}$, and then aggregating them with function $\mathcal{A}$ followed by a nonlinear activator $\sigma$. In this formulation, the two functions $\mathcal{A}$ and $\mathcal{T}$ are the key to $\mathbf{f}_{P_{\text{sub}}}$. That is, the permutation invariance of point set can be achieved only when $\mathcal{A}$ is symmetric (*e.g.*, summation) and $\mathcal{T}$ is shared over each point in $\mathcal{N}(x_i)$.

**Limitations of classic CNN.** In classic CNN, $\mathcal{T}$ is implemented as $\mathcal{T}(\mathbf{f}_{x_j}) = \mathbf{w}_j \cdot \mathbf{f}_{x_j}$, where $\mathbf{w}_j$ is learnable weight and "$\cdot$" denotes element-wise multiplication. There are mainly two limitations of this convolution when applied on point cloud: 1) $\mathbf{w}_j$ is not shared over each point in $\mathcal{N}(x_i)$, resulting in variance to point permutation and incapability to process irregular $P_{\text{sub}}$ (*e.g.*, different number); 2) the gradient of $\mathbf{w}_j$ in backpropagation is only relevant to the isolated point $x_j$, leading to an implicit learning strategy, which could not bring much shape awareness and robustness to $\mathbf{f}_{P_{\text{sub}}}$. This issue can be partly alleviated by some techniques like performing various data augmentations or using lots of convolutional filters, yet they are suboptimal.

**Conversion: Learn from relation.** We argue that the above limitations can be mitigated by learning from relation. In the neighborhood of 3D space, the geometric relation between $x_i$ and all its neighbors $\mathcal{N}(x_i)$ is an explicit expression about the spatial layout of points, which further discriminatively reflects the underlying shape. To capture this relation, we replace $\mathbf{w}_j$ in classical CNN with $\mathbf{w}_{ij}$, which learns a mapping $\mathcal{M}$ of a relation vector $\mathbf{h}_{ij}$, *i.e.*, the predefined geometric priors between $x_i$ and $x_j$. We call $\mathbf{h}_{ij}$ as low-level relation. This process can be described as

$$\mathcal{T}(\mathbf{f}_{x_j}) = \mathbf{w}_{ij}\cdot\mathbf{f}_{x_j} = \mathcal{M}(\mathbf{h}_{ij})\cdot\mathbf{f}_{x_j}. \tag{2}$$

The goal of mapping $\mathcal{M}$ is to abstract high-level relation expression between two points, which can encode their spatial layout. Here we implement $\mathcal{M}$ with a shared multi-layer perceptron (MLP) due to its powerful mapping ability. This process is illustrated in the middle part of Fig. 2. In this way, $\mathbf{w}_j$ is neatly converted to $\mathbf{w}_{ij}$, whose gradient (determined by $\mathbf{h}_{ij}$) is relevant to both $x_i$ and $x_j$. Meanwhile, $\mathcal{M}$ is exactly shared over all the points in $\mathcal{N}(x_i)$, making it independent to the irregularity of points. It can also be robust to rigid transformation that will be clarified in Sec 3.2.

As a consequence, $\mathbf{f}_{P_{\text{sub}}}$ in Eq. (1) becomes

$$\mathbf{f}_{P_{\text{sub}}} = \sigma\big(\mathcal{A}(\{\mathcal{M}(\mathbf{h}_{ij})\cdot\mathbf{f}_{x_j},\ \forall x_j\})\big). \tag{3}$$

This convolutional representation, with all the relation between $x_i$ and $\mathcal{N}(x_i)$ aggregated, can achieve explicit reasoning about the spatial layout of points, thus resulting in discriminative shape awareness. For geometric priors, one can use 3D Euclidean distance as an intuitive description of low-level relation $\mathbf{h}_{ij}$. Moreover, $\mathbf{h}_{ij}$ can also be defined flexibly since $\mathcal{M}$ can map it to a high-dimensional relation vector for channel alignment with $\mathbf{f}_{x_j}$ for easy multiplication. We will discuss $\mathbf{h}_{ij}$ in detail in the experiment section.

[1] In this paper, the bias term is omitted for clarity.

**Channel-raising mapping.** In Eq. (3), the channel number of $\mathbf{f}_{P_{\text{sub}}}$ is the same as the input feature $\mathbf{f}_{x_j}$. This is inconsistent with classic image CNN that increases channel number while decreasing image resolution for a more abstract representation. For example, the channel number of 64-128-256-512 is set in VGG network [32]. Accordingly, we add a shared MLP on $\mathbf{f}_{P_{\text{sub}}}$ for further channel-raising mapping. It is illustrated in the middle part of Fig. 2.

### 3.2. Properties

RS-Conv in Eq. (3) can maintain four decent properties:

**Permutation invariance.** In the inner mapping function $\mathcal{M}(\mathbf{h})$, both the low-level relation $\mathbf{h}$ and the shared MLP $\mathcal{M}$ are invariant to the input order of points. Therefore, with the outer aggregation function $\mathcal{A}$ being symmetric, the permutation invariance can be satisfied.

**Robustness to rigid transformation.** This property is well held in the high-level relation encoding $\mathcal{M}(\mathbf{h})$. It can be robust to rigid transformation, *e.g.*, translation and rotation, when a suitable $\mathbf{h}$ (*e.g.*, 3D Euclidean distance) is defined.

**Points interaction.** Points are not isolated and nearby points form a meaningful shape in geometric space. Thus their inherent interaction is critical for discriminative shape awareness. Our solution of relation learning explicitly encode the geometric relation among points, naturally capturing the interaction of points.

**Weight sharing.** This is the key property that allows applying the same learning function over different irregular point subsets for robustness, as well as low complexity. In Eq. (3), the symmetric $\mathcal{A}$, the shared MLP $\mathcal{M}$ and the predefined geometric priors $\mathbf{h}$ are all independent to the irregularity of points. Hence, this property is also satisfied.

### 3.3. Revisiting 2D Grid Convolution

The proposed RS-Conv is a generic formulation of 2D grid convolution for relation reasoning. We clarify this with a neighborhood (convolution kernel) of $3 \times 3$ on a 2D-grid feature map, as illustrated in Fig. 3. Specifically, the summation function $\sum$ is a specific instance of the aggregation function $\mathcal{A}$. Moreover, note that $w_j$ always implies a fixed positional relation between $x_i$ and its neighbor $x_j$ in the regular grid. For example, $w_1$ always implies the top-left relation with $x_i$, and $w_2$ implies the right-above relation with $x_i$. In other words, $w_j$ is actually constrained to encode one kind of regular grid relation in the learning process. Therefore, our RS-Conv with relation learning is more general and can be applied to model 2D grid spatial relationship.

### 3.4. RS-CNN for Point Cloud Analysis

Using RS-Conv (Fig. 2) as a basic operator and adopting a uniform sampling strategy, a hierarchical shape-aware

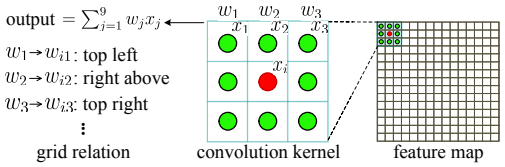


Figure 3. Illustration of 2D grid convolution with a kernel of $3 \times 3$.

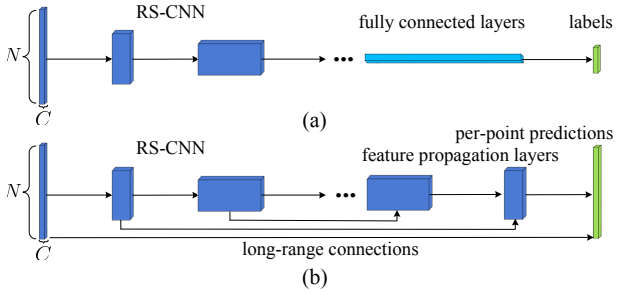


Figure 4. The architectures of RS-CNN applied in the classification (a) and segmentation (b) of point cloud. $N$ is the number of points and $C$ is the channel number.

learning architecture like classic CNN, namely, RS-CNN, can be developed for point cloud analysis as

$$\mathbf{F}^{\ell}_{P_{N_\ell}} = \text{RS-CONV}(\mathbf{F}^{\ell-1}_{P_{N_{\ell-1}}}), \tag{4}$$

where $\mathbf{F}^{\ell}_{P_{N_\ell}}$, features in layer $\ell$ of the sampled point set $P_{N_\ell}$ with number $N_\ell$, are obtained by applying RS-Conv on the features in the previous layer $\ell - 1$.

Our RS-CNN applied in the classification and segmentation of point cloud is illustrated in Fig. 4. In both tasks, RS-CNN is used for learning a group of hierarchical shape-aware representation. The final global representation followed by three fully connected (FC) layers is configured for classification. For segmentation, the learned multi-level representation is successively upsampled by feature propagation [26] to generate per-point predictions. Both of them can be trained in an end-to-end manner.

### 3.5. Implementation Details

**RS-Conv in Eq. (3).** Symmetric function max pooling is applied as aggregation function $\mathcal{A}$. ReLU [23] is used as nonlinear activator $\sigma$. For mapping function $\mathcal{M}$, a three-layer shared MLP is deployed since theoretically it can fit arbitrary continuous mappings [8]. Low-level relation $\mathbf{h}_{ij}$ is defined as a compact vector with 10 channels, *i.e.*, (3D Euclidean distance, $x_i - x_j$, $x_i$, $x_j$). The channel-raising mapping is achieved by a single-layer shared MLP. Batch normalization [12] is applied in each MLP.

**RS-CNN for points analysis.** The farthest points are picked from point cloud for sampling local subsets to perform RS-Conv. In each neighborhood, a fixed number of neighbors are randomly sampled for batch processing, and

they are normalized to take the centroid as the origin. To capture more sufficient geometric relation, we force RS-CNN to learn over three-scale neighborhoods centered on a sampled point with a shared weight. This is different from multi-scale grouping (MSG) [26] that learns multi-scale features using multiple groups of weight. RS-CNN with 3 layers and 4 layers is deployed for classification and segmentation, respectively. Note that only 3D coordinates xyz are used as the input features to RS-CNN.

Our RS-CNN is implemented using Pytorch[2]. The Adam optimization algorithm is employed for training, with a mini-batch size of 32. The momentum for BN starts with 0.9 and decays with a rate of 0.5 every 20 epochs. The learning rate begins with 0.001 and decays with a rate of 0.7 every 20 epochs. The weight of RS-CNN is initialized using the techniques introduced by He *et al*. [7].

## 4. Experiment

In this section, we arrange comprehensive experiments to validate the proposed RS-CNN. First, we evaluate RS-CNN for point cloud analysis on three tasks (Sec 4.1). We then provide detailed experiments to carefully study RS-CNN (Sec 4.2). Finally, we visualize the shape features that RS-CNN captures and analyze the complexity (Sec 4.3).

### 4.1. Point Cloud Analysis

**Shape classification.** We evaluate RS-CNN on ModelNet40 classification benchmark [42]. It is composed of 9843 train models and 2468 test models in 40 classes. The point cloud data is sampled from these models by [24]. We uniformly sample 1024 points and normalize them to a unit sphere. During training, we augment the input data with random anisotropic scaling in the range [-0.66, 1.5] and translation in the range [-0.2, 0.2], as in [16]. Meanwhile, dropout technique [33] with 50% ratio is applied in FC layers. During testing, similar to [24, 26], we perform ten voting tests with random scaling and average the predictions.

The quantitative comparisons with the state-of-the-art point-based methods are summarized in Table 1, where RS-CNN outperforms all the xyz-input methods. Specifically, RS-CNN reduces the error rate of PointNet++ [26] by 31.2%, and surpasses its advanced version that uses additional normal data as well as very dense points (5k). Moreover, even using only xyz as the input, RS-CNN can also achieve a superior result (93.6%) compared with the best additional-input method SO-Net [19] (93.4%). This convincingly verifies the effectiveness of our RS-CNN.

We test the robustness of RS-CNN on sampling density, by using sparser points of number 1024, 512, 256, 128 and 64 as the input to a model trained with 1024 points. As in [26], random input dropout technique is applied for a fair comparison. Fig. 5 shows the test results, where

[2]https://github.com/Yochengliu/Relation-Shape-CNN

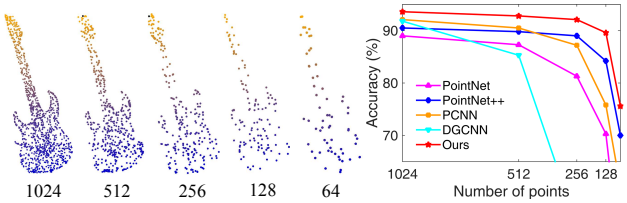


Figure 5. Left part: Point cloud with random point dropout. Right part: Test results of using sparser points as the input to a model trained with 1024 points.

Table 1. Shape classification results (%) on ModelNet40 benchmark (nor: normal, "-": unknown).

| method | input | #points | acc. |
|---|---|---|---|
| Pointwise-CNN [10] | xyz | 1k | 86.1 |
| Deep Sets [48] | xyz | 1k | 87.1 |
| ECC [31] | xyz | 1k | 87.4 |
| PointNet [24] | xyz | 1k | 89.2 |
| SCN [44] | xyz | 1k | 90.0 |
| Kd-Net(depth=10) [16] | xyz | 1k | 90.6 |
| PointNet++ [26] | xyz | 1k | 90.7 |
| KCNet [30] | xyz | 1k | 91.0 |
| MRTNet [3] | xyz | 1k | 91.2 |
| Spec-GCN [38] | xyz | 1k | 91.5 |
| PointCNN [21] | xyz | 1k | 91.7 |
| DGCNN [41] | xyz | 1k | 92.2 |
| PCNN [1] | xyz | 1k | 92.3 |
| **Ours** | **xyz** | **1k** | **93.6** |
| SO-Net [19] | xyz | 2k | 90.9 |
| Kd-Net(depth=15) [16] | xyz | 32k | 91.8 |
| O-CNN [39] | xyz, nor | - | 90.6 |
| Spec-GCN [38] | xyz, nor | 1k | 91.8 |
| PointNet++ [26] | xyz, nor | 5k | 91.9 |
| SpiderCNN [45] | xyz, nor | 5k | 92.4 |
| SO-Net [19] | xyz, nor | 5k | 93.4 |

the compared methods are PointNet [24], PointNet++ [26], PCNN [1] and DGCNN [41]. As can be seen, it is more difficult for shape recognition when points get sparser. Even so, RS-CNN is still considerably robust. It achieves nearly consistent robustness as PointNet++, whilst showing superior performance on each density.

**Shape part segmentation.** Part segmentation is a challenging task for fine-grained shape analysis. We evaluate RS-CNN for this task on ShapeNet part benchmark [46] and follow the data split in [24]. This dataset contains 16881 shapes with 16 categories, and is labeled in 50 parts in total. As in [24], we randomly pick 2048 points as the input and concatenate the one-hot encoding of the object label to the last feature layer. During testing, we also apply ten voting tests using random scaling. Except for standard IoU (*Inter-over-Union*) on each category, we also report two types of mean IoU (mIoU) that are averaged across all classes and all instances, respectively.

Table 2. Shape part segmentation results (%) on ShapeNet part benchmark (nor: normal, "-": unknown).

| method | input | class mIoU | instance mIoU | air plane | bag | cap | car | chair | ear phone | guitar | knife | lamp | laptop | motor bike | mug | pistol | rocket | skate board | table |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Kd-Net [16] | 4k | 77.4 | 82.3 | 80.1 | 74.6 | 74.3 | 70.3 | 88.6 | 73.5 | 90.2 | 87.2 | 81.0 | 94.9 | 57.4 | 86.7 | 78.1 | 51.8 | 69.9 | 80.3 |
| PointNet [24] | 2k | 80.4 | 83.7 | 83.4 | 78.7 | 82.5 | 74.9 | 89.6 | 73.0 | 91.5 | 85.9 | 80.8 | 95.3 | 65.2 | 93.0 | 81.2 | 57.9 | 72.8 | 80.6 |
| RS-Net [11] | - | 81.4 | 84.9 | 82.7 | **86.4** | 84.1 | 78.2 | 90.4 | 69.3 | 91.4 | 87.0 | 83.5 | 95.4 | 66.0 | 92.6 | 81.8 | 56.1 | 75.8 | 82.2 |
| SCN [44] | 1k | 81.8 | 84.6 | 83.8 | 80.8 | 83.5 | 79.3 | 90.5 | 69.8 | **91.7** | 86.5 | 82.9 | 96.0 | 69.2 | 93.8 | 82.5 | **62.9** | 74.4 | 80.8 |
| PCNN [1] | 2k | 81.8 | 85.1 | 82.4 | 80.1 | 85.5 | 79.5 | 90.8 | 73.2 | 91.3 | 86.0 | 85.0 | 95.7 | 73.2 | 94.8 | 83.3 | 51.0 | 75.0 | 81.8 |
| SPLATNet [34] | - | 82.0 | 84.6 | 81.9 | 83.9 | 88.6 | 79.5 | 90.1 | 73.5 | 91.3 | 84.7 | 84.5 | **96.3** | 69.7 | **95.0** | 81.7 | 59.2 | 70.4 | 81.3 |
| KCNet [30] | 2k | 82.2 | 84.7 | 82.8 | 81.5 | 86.4 | 77.6 | 90.3 | 76.8 | 91.0 | 87.2 | 84.5 | 95.5 | 69.2 | 94.4 | 81.6 | 60.1 | 75.2 | 81.3 |
| DGCNN [41] | 2k | 82.3 | 85.1 | **84.2** | 83.7 | 84.4 | 77.1 | 90.9 | 78.5 | 91.5 | 87.3 | 82.9 | 96.0 | 67.8 | 93.3 | 82.6 | 59.7 | 75.5 | 82.0 |
| **Ours** | **2k** | **84.0** | **86.2** | 83.5 | 84.8 | **88.8** | **79.6** | **91.2** | **81.1** | 91.6 | **88.4** | **86.0** | 96.0 | **73.7** | 94.1 | **83.4** | 60.5 | **77.7** | **83.6** |
| PointNet++ [26] | 2k,nor | 81.9 | 85.1 | 82.4 | 79.0 | 87.7 | 77.3 | 90.8 | 71.8 | 91.0 | 85.9 | 83.7 | 95.3 | 71.6 | 94.1 | 81.3 | 58.7 | 76.4 | 82.6 |
| SyncCNN [47] | mesh | 82.0 | 84.7 | 81.6 | 81.7 | 81.9 | 75.2 | 90.2 | 74.9 | 93.0 | 86.1 | 84.7 | 95.6 | 66.7 | 92.7 | 81.6 | 60.6 | 82.9 | 82.1 |
| SO-Net [19] | 1k,nor | 80.8 | 84.6 | 81.9 | 83.5 | 84.8 | 78.1 | 90.8 | 72.2 | 90.1 | 83.6 | 82.3 | 95.2 | 69.3 | 94.2 | 80.0 | 51.6 | 72.1 | 82.6 |
| SpiderCNN [45] | 2k,nor | 82.4 | 85.3 | 83.5 | 81.0 | 87.2 | 77.5 | 90.7 | 76.8 | 91.1 | 87.3 | 83.3 | 95.8 | 70.2 | 93.5 | 82.7 | 59.7 | 75.8 | 82.8 |

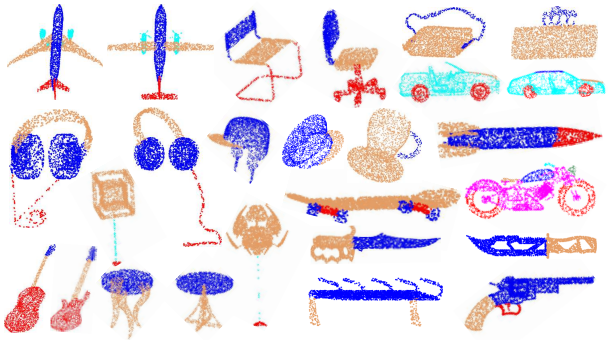

Figure 6. Segmentation examples on ShapeNet part benchmark.

Table 2 summarizes the quantitative comparisons with the state-of-the-art methods, where RS-CNN achieves the best performance with class mIoU of 84.0% and instance mIoU of 86.2%. This considerably surpasses the second best xyz-based methods, *i.e.*, DGCNN [41] with 82.3% (1.7↑) in class mIoU and PCNN [1] with 85.1% (1.1↑) in instance mIoU, respectively. Noticeably, RS-CNN sets new state of the arts in the xyz-based methods over ten categories. These improvements demonstrate the robustness of RS-CNN to diverse shape structures. Fig. 6 shows some segmentation examples. One can see that although the part shapes implied in irregular points are varied and they may be very confusing to recognize, RS-CNN can also segment them out with decent accuracy.

**Normal estimation.** Normal estimation in point cloud is a crucial step for numerous applications, such as surface reconstruction and rendering. This task is very challenging since it requires a higher level of reasoning, which goes beyond the underlying shape recognition. We take normal estimation as a supervised regression task, and achieve it using the segmentation network. The cosine-loss between the normalized output and ground truth normal is applied for regression training. ModelNet40 dataset is used for evaluation, with uniformly sampled 1024 points as the input.

The quantitative results are summarized in Table 3. RS-CNN outperforms other advanced methods on this task with

Table 3. Normal estimation error on ModelNet40 dataset.

| dataset | method | #points | error |
|---|---|---|---|
| ModelNet40 | PointNet [1] | 1k | 0.47 |
| | PointNet++ [1] | 1k | 0.29 |
| | PCNN [1] | 1k | 0.19 |
| | **Ours** | **1k** | **0.15** |

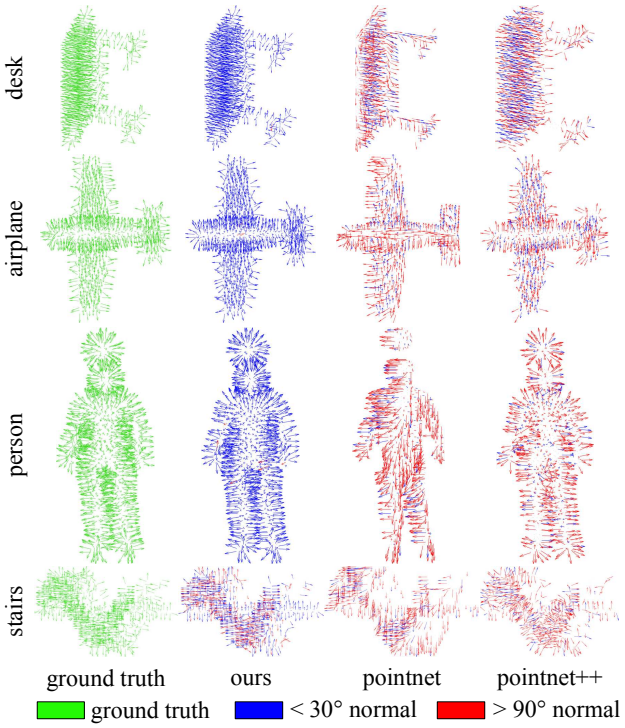


Figure 7. Normal estimation on ModelNet40 dataset. For clearness, we only show predictions with angle less than 30° in blue, and angle greater than 90° in red between ground truth normals.

a lower error of 0.15. This significantly reduces the error of PointNet++ (0.29) by 48.3%. Fig. 7 shows some normal estimation examples, where our RS-CNN with geometric relation learning can obtain more decent predictions. However, RS-CNN could also be less effective for some intractable shapes, such as spiral stairs and intricate plants.

Table 4. Ablation study of RS-CNN (%). "DP" indicates the dropout technique in FC layers of the classification network.

| model | #points | relation | BN | DP | scale | voting | acc. |
|---|---|---|---|---|---|---|---|
| A | 1k | | | | 1 | | 87.2 |
| B | 1k | ✓ | | | 1 | | 89.9 |
| C | 1k | ✓ | ✓ | | 1 | | 91.9 |
| D | 1k | ✓ | ✓ | ✓ | 1 | | 92.2 |
| E | 1k | ✓ | ✓ | ✓ | 2 | | 92.5 |
| F | 1k | ✓ | ✓ | ✓ | 3 | | 92.9 |
| G | 1k | ✓ | ✓ | ✓ | 3 | ✓ | **93.6** |
| H | 2k | ✓ | ✓ | ✓ | 3 | ✓ | **93.6** |
| I | 1k | | ✓ | ✓ | 3 | ✓ | 90.1 |

## 4.2. RS-CNN Design Analysis

In this section, we first perform a detailed ablation study on RS-CNN. Then, we discuss the choices of aggregation function $\mathcal{A}$, mapping function $\mathcal{M}$ and low-level relation $\mathbf{h}$ in Eq. (3). Finally, we validate the robustness of RS-CNN on point permutation and rigid transformation. All experiments are conducted on ModelNet40 classification dataset.

**Ablation study.** The results are summarized in Table 4. The baseline (model A) is set to learn without geometric relation encoding, but with a shared three-layer MLP as feature transformation function $\mathcal{T}$ in Eq. (1).

The baseline only gets an accuracy of 87.2%. Yet with geometric relation learning, it is significantly improved to 89.9% (model B). This convincingly verifies the effectiveness of our RS-CNN. Then, a great improvement of 2% is gained after using BN (model C), maybe because it can greatly ease the network training. Moreover, dropout technique improves the result by 0.3% (model D). As mentioned in Sec 3.5, RS-CNN should be able to benefit from sufficient geometric relation. This is verified by model E (92.5%) and model F (92.9%) that perform two-scale and three-scale relation learning, respectively. Eventually, with ten voting tests, an impressive accuracy of 93.6% (model G) can be obtained with only xyz features.

To investigate the impact of the number of input points on RS-CNN, we also train the network with 2048 points but find no improvement (model H). In addition, to compare with the baseline (model A) more fairly, we set a new baseline (model I) that works with all the techniques but relation learning. It gets an accuracy of 90.1%, which RS-CNN can also surpass by 3.5%. We speculate that RS-CNN with geometric relation reasoning can acquire more discriminative shape awareness, and this awareness can be greatly enhanced by multi-scale relation learning.

**Aggregation function $\mathcal{A}$.** Three symmetric functions: max pooling (max), average pooling (avg.) and summation (sum), are employed to study the effect of $\mathcal{A}$ on RS-CNN. Table 5 summarizes the results. As can be seen, with $\mathcal{M}$ using three layers, max pooling achieves the best performance while average pooling and summation get the same accuracy. The reason may be that max pooling can select the biggest feature response, thus keeping the most expressive representation and removing redundant information.

Table 5. The results (%) of different designs on aggregation function $\mathcal{A}$ and mapping function $\mathcal{M}$ (Eq. (3)) ($\mathcal{M}_{(k)}$: $k$-layer MLP).

| $\mathcal{A}$ | $\mathcal{M}_{(2)}$ | $\mathcal{M}_{(3)}$ | $\mathcal{M}_{(4)}$ | acc. |
|---|---|---|---|---|
| max | ✓ | | | 92.4 |
| max | | ✓ | | **93.6** |
| max | | | ✓ | 92.7 |
| avg. | | ✓ | | 91.6 |
| sum | | ✓ | | 91.6 |

Table 6. The results (%) of five intuitive low-level relations $\mathbf{h}$ (Ed: Euclidean distance, cosd: cosine distance, $x^{\text{nor}}$: normal of $x$, $x'$: 2D projection of $x$). Model A applies only 3D Euclidean distance as $\mathbf{h}$; Model B adds the coordinates difference to model A; Model C adds the coordinates of two points to model B; Model D utilizes the normals of two points and their cosine distance as $\mathbf{h}$; Model E projects 3D points onto a 2D plane of XY, XZ and YZ.

| model | low-level relation $\mathbf{h}$ | channels | acc. |
|---|---|---|---|
| A | (3D-Ed) | 1 | 92.5 |
| B | (3D-Ed, $x_i - x_j$) | 4 | 93.0 |
| C | (3D-Ed, $x_i - x_j$, $x_i$, $x_j$) | 10 | **93.6** |
| D | (3D-cosd, $x_i^{\text{nor}}$, $x_j^{\text{nor}}$) | 7 | 92.8 |
| E | (2D-Ed, $x_i' - x_j'$, $x_i'$, $x_j'$) | 10 | $\approx$ 92.2 |

**Mapping function $\mathcal{M}$.** The results of $\mathcal{M}$ deployed with different layers are summarized in the first three rows of Table 5. One can see that the best accuracy of 93.6% is obtained by a shared three-layer MLP, and it decreases by 0.9% when increasing the number of layers. The reason might be that $\mathcal{M}$ with four layers brings some difficulty for network training. Noticeably, RS-CNN can also get a decent accuracy of 92.4% with $\mathcal{M}$ using only two layers. This verifies the powerfulness of relation learning for underlying shape capturing from point cloud.

**Low-level relation $\mathbf{h}$.** The key to RS-CNN is learning from relation, thus how to define $\mathbf{h}$ is an issue worth exploring. Actually, $\mathbf{h}$ can be defined flexibly, as long as it could discriminatively reflect the underlying shape. To validate this claim and facilitate the understanding, we experiment with five intuitive relation definitions as examples, whose results are summarized in Table 6.

As can be seen, using only 3D Euclidean distance as $\mathbf{h}$, the accuracy can also reach 92.5% (model A). This demonstrates the effectiveness of our RS-CNN for high-level geometric relation learning. Moreover, the performance is gradually improved with additional relation, including coordinates difference (model B) and coordinates themselves (model C). We also utilize the normal vectors of two points and their cosine distance as $\mathbf{h}$, the result (model D) is 92.8%. This indicates RS-CNN is also able to abstract shape information from the relation in normals.

Table 7. Robustness to point permutation and rigid transformation (%). During testing, we perform random permutation (perm.) of points, add a small translation of $\pm 0.2$ and counterclockwise rotate the input point cloud by 90° and 180° around Y axis.

| method | acc. | perm. | +0.2 | -0.2 | 90° | 180° |
|---|---|---|---|---|---|---|
| PointNet [24] | 88.7 | 88.7 | 70.8 | 70.6 | 42.5 | 38.6 |
| PointNet++ [26] | 88.2† | 88.2 | 88.2 | 88.2 | 47.9 | 39.7 |
| **Ours** | **90.3**† | **90.3** | **90.3** | **90.3** | **90.3** | **90.3** |

† The accuracy drops a lot mainly because the forcible normalization of each local point subset could bring difficulty for shape recognition.

Intuitively, the relation among points in the 2D view of point cloud can also reflect the underlying shape. Therefore, to validate our RS-CNN for shape abstraction on 2D relation, we forcibly set the value of one dimension in 3D coordinates to be zero, *i.e.*, projecting 3D points onto a 2D plane of XY, XZ and YZ. The results are all around 92.2% (model E), which is quite impressive. This further verifies the effectiveness of the proposed relation learning method.

**Robustness to point permutation and rigid transformation.** We compare the robustness of our RS-CNN with PointNet [24] and PointNet++ [26]. Note that all the models are trained without related data augmentations, *e.g.*, translation or rotation, to avoid confusion in this test. In addition, although relation learning in RS-CNN is robust to rotation, the initial input features of 3D coordinates are affected. We address this issue by normalizing each sampled point subset to corresponding local coordinate system, which is determined by each sampled point and its normal. For a fair comparison, we also perform this normalization for PointNet++, as it learns over local subsets as well. The 3D Euclidean distance is applied as geometric relation $\mathbf{h}$ in RS-CNN for this test. Table 7 summarizes the test results.

As can be seen, all the methods are invariant to permutation. However, PointNet is vulnerable to both translation and rotation while PointNet++ is sensitive to rotation. By contrast, our RS-CNN with geometric relation learning is invariant to these perturbations, making it powerful for robust shape recognition.

### 4.3. Visualization and Complexity Analysis

**Visualization.** Fig. 8 visualizes the shape features learned by the first two layers of RS-CNN on ModelNet40 dataset. As it shows, the features learned by the first layer mostly respond to edges, corners and arcs, while the ones in the second layer capture more semantical shape parts like airfoils and heads. This verifies RS-CNN can learn progressive shape-aware representation for point cloud analysis.

**Complexity Analysis.** Table 8 summarizes the space (number of params) and the time (floating point operations/sample) complexity of RS-CNN in classification with 1024 points as the input. Compared with PointNet [24], RS-CNN reduces the params by 59.7% and the FLOPs by 32.9%, which shows its great potential for real-time applications, *e.g.*, scene parsing in autonomous driving.

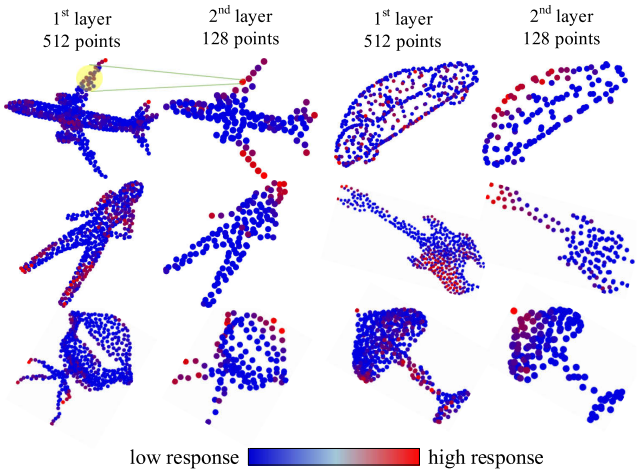


Figure 8. Visualization of the shape features learned by the first two layers of RS-CNN on ModelNet40 dataset. The features learned by the first layer mostly respond to edges, corners and arcs, while the ones in the second layer capture more semantical shape parts like airfoils and heads.

Table 8. Complexity of RS-CNN in point cloud classification.

| method | #params | #FLOPs/sample |
|---|---|---|
| PointNet [24] | 3.50M | 440M |
| PointNet++ [21] | 1.48M | 1684M |
| PCNN [21] | 8.20M | **294M** |
| **Ours** | **1.41M** | 295M |

## 5. Conclusion

In this work, RS-CNN, namely, Relation-Shape Convolutional Neural Network, which extends regular grid CNN to irregular configuration for point cloud analysis, has been proposed. The core to RS-CNN is a novel convolution operator, which learns from relation, *i.e.*, the geometric topology constraint among points. In this way, explicit reasoning about the spatial layout of points can be made to obtain discriminative shape awareness. Moreover, the decent properties of geometric relation can also be acquired, such as robustness to rigid transformation. As a consequence, RS-CNN equipped with this operator can achieve contextual shape-aware learning, making it highly effective. Extensive experiments on challenging benchmarks across three tasks, as well as thorough empirical and theoretical analysis, have demonstrated RS-CNN achieves the state of the arts.

## Acknowledgement

The authors thank anonymous reviewers very much for their valuable comments that greatly improve this paper. This work is supported by the National Natural Science Foundation of China under Grants 61573352, 91646207 and 61773377, the Young Elite Scientists Sponsorship Program by CAST under Grant 2018QNRC001 and the Beijing Natural Science Foundation under Grant L172053.

## References


[1] M. Atzmon, H. Maron, and Y. Lipman. Point convolutional neural networks by extension operators. In *SIGGRAPH*, pages 1–14, 2018. 2, 5, 6

[2] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. Gao. GVCNN: Group-view convolutional neural networks for 3D shape recognition. In *CVPR*, pages 264–272, 2018. 1, 2

[3] M. Gadelha, R. Wang, and S. Maji. Multiresolution tree networks for 3D point cloud processing. In *ECCV*, pages 105–122, 2018. 1, 2, 5

[4] P. Guerrero, Y. Kleiman, M. Ovsjanikov, and N. J. Mitra. PCPNet: Learning local shape properties from raw point clouds. *Comput. Graph. Forum*, 37(2):75–85, 2018. 2

[5] H. Guo, J. Wang, Y. Gao, J. Li, and H. Lu. Multi-view 3D object retrieval with deep embedding network. *IEEE Trans. Image Processing*, 25(12):5526–5537, 2016. 1, 2

[6] Z. Han, M. Shang, Z. Liu, C. Vong, Y. Liu, M. Zwicker, J. Han, and C. L. P. Chen. SeqViews2SeqLabels: Learning 3D global features via aggregating sequential views by RNN with attention. *IEEE Trans. Image Processing*, 28(2):658–672, 2019. 2

[7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *ICCV*, pages 1026–1034, 2015. 5

[8] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. 4

[9] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei. Relation networks for object detection. In *CVPR*, pages 3588–3597, 2018. 2

[10] B.-S. Hua, M.-K. Tran, and S.-K. Yeung. Pointwise convolutional neural networks. In *CVPR*, pages 974–993, 2018. 5

[11] Q. Huang, W. Wang, and U. Neumann. Recurrent slice networks for 3D segmentation of point clouds. In *CVPR*, pages 2626–2635, 2018. 6

[12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 4

[13] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NeurIPS*, pages 2017–2025, 2015. 2

[14] V. Jampani, M. Kiefel, and P. V. Gehler. Learning sparse high dimensional filters: Image filtering, dense CRFs and bilateral neural networks. In *CVPR*, pages 4452–4461, 2016. 2

[15] M. Jiang, Y. Wu, and C. Lu. PointSIFT: A SIFT-like network module for 3D point cloud semantic segmentation. *arXiv preprint arXiv:1807.00652*, 2018. 1

[16] R. Klokov and V. S. Lempitsky. Escape from cells: Deep Kd-Networks for the recognition of 3D point cloud models. In *ICCV*, pages 863–872, 2017. 2, 5, 6

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NeurIPS*, pages 1106–1114, 2012. 1, 3

[18] L. Landrieu and M. Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *CVPR*, pages 4558–4567, 2018. 1, 2

[19] J. Li, B. M. Chen, and G. H. Lee. SO-Net: Self-organizing network for point cloud analysis. In *CVPR*, pages 9397–9406, 2018. 2, 5, 6

[20] R. Li, S. Wang, F. Zhu, and J. Huang. Adaptive graph convolutional neural networks. In *AAAI*, pages 3546–3553, 2018. 2

[21] Y. Li, R. Bu, M. Sun, and B. Chen. PointCNN: Convolution on X-transformed points. In *NeurIPS*, pages 828–838, 2018. 2, 5, 8

[22] D. Maturana and S. Scherer. VoxNet: A 3D convolutional neural network for real-time object recognition. In *IROS*, pages 922–928, 2015. 1, 2

[23] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010. 4

[24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, pages 77–85, 2016. 1, 2, 5, 6, 8

[25] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view CNNs for object classification on 3D data. In *CVPR*, pages 5648–5656, 2016. 2

[26] C. R. Qi, L. Yi, H. su, and L. J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, pages 5099–5108, 2017. 1, 2, 4, 5, 6, 8

[27] S. Ravanbakhsh, J. Schneider, and B. Poczos. Deep learning with sets and point clouds. In *ICLR*, pages 1–12, 2017. 1, 2

[28] G. Riegler, A. O. Ulusoy, and A. Geiger. OctNet: Learning deep 3D representations at high resolutions. In *CVPR*, pages 6620–6629, 2017. 2

[29] A. Savchenkov. Generalized convolutional neural networks for point cloud data. In *ICMLA*, pages 930–935, 2017. 1

[30] Y. Shen, C. Feng, Y. Yang, and D. Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *CVPR*, pages 4548–4557, 2018. 2, 5, 6

[31] M. Simonovsky and N. Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *CVPR*, pages 29–38, 2017. 5

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, pages 1–14, 2015. 1, 3, 4

[33] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research.*, 15(1):1929–1958, 2014. 5

[34] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and J. Kautz. SPLATNet: Sparse lattice networks for point cloud processing. In *CVPR*, pages 2530–2539, 2018. 1, 2, 6

[35] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller. Multi-view convolutional neural networks for 3D shape recognition. In *ICCV*, pages 945–953, 2015. 1, 2

[36] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. In *ICCV*, pages 2107–2115, 2017. 2

[37] G. Te, W. Hu, A. Zheng, and Z. Guo. RGCNN: Regularized graph CNN for point cloud segmentation. In *MM*, pages 746–754, 2018. 2

[38] C. Wang, B. Samari, and K. Siddiqi. Local spectral graph convolution for point set feature learning. In *ECCV*, pages 1–16, 2018. 1, 2, 5

[39] P. Wang, Y. Liu, Y. Guo, C. Sun, and X. Tong. O-CNN: octree-based convolutional neural networks for 3D shape analysis. *ACM Trans. Graph.*, 36(4):72:1–72:11, 2017. 2, 5

[40] X. Wang, R. B. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, pages 7794–7803, 2018. 2

[41] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph CNN for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 2, 5, 6

[42] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D ShapeNets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 1, 2, 5

[43] J. Xie, G. Dai, F. Zhu, E. K. Wong, and Y. Fang. DeepShape: Deep-learned shape descriptor for 3D shape retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(7):1335–1345, 2017. 2

[44] S. Xie, S. Liu, Z. Chen, and Z. Tu. Attentional ShapeContextNet for point cloud recognition. In *CVPR*, pages 4606–4615, 2018. 5, 6

[45] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao. SpiderCNN: Deep learning on point sets with parameterized convolutional filters. In *ECCV*, pages 90–105, 2018. 1, 5, 6

[46] L. Yi, V. G. Kim, D. Ceylan, I. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. J. Guibas. A scalable active framework for region annotation in 3D shape collections. *ACM Trans. Graph.*, 35(6):210:1–210:12, 2016. 5

[47] L. Yi, H. Su, X. Guo, and L. J. Guibas. SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation. In *CVPR*, pages 6584–6592, 2017. 6

[48] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *NeurIPS*, pages 3394–3404, 2017. 1, 2, 5

[49] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014. 1