3-2022

RIConv++: Effective rotation invariant convolutions for 3D point clouds deep learning

Zhiyuan ZHANG
Singapore Management University, zhiyuanzhang@smu.edu.sg

Binh-Son HUA

Sai-Kit YEUNG

Noname manuscript No.
(will be inserted by the editor)

RIConv++: Effective Rotation Invariant Convolutions for 3D Point Clouds Deep Learning

Zhiyuan Zhang · Binh-Son Hua · Sai-Kit Yeung

Received: date / Accepted: date

Abstract 3D point clouds deep learning is a promising field of research that allows a neural network to learn features of point clouds directly, making it a robust tool for solving 3D scene understanding tasks. While recent works show that point cloud convolutions can be invariant to translation and point permutation, investigations of the rotation invariance property for point cloud convolution has been so far scarce. Some existing methods perform point cloud convolutions with rotation-invariant features, existing methods generally do not perform as well as translation-invariant only counterpart. In this work, we argue that a key reason is that compared to point coordinates, rotation-invariant features consumed by point cloud convolution are not as distinctive. To address this problem, we propose a simple yet effective convolution operator that enhances feature distinction by designing powerful rotation invariant features from the local regions. We consider the relationship between the point of interest and its neighbors as well as the internal relationship of the neighbors to largely improve the feature descriptiveness. Our network architecture can capture both local and global context by simply tuning the neighborhood size in each convolution layer. We conduct several experiments on synthetic and real-world point cloud classifications, part segmentation, and shape retrieval to evaluate our method, which achieves the state-of-the-art accuracy under challenging rotations.

Keywords 3D Point Cloud · Convolutional Neural Networks · Deep Learning · Rotation Invariance

Zhiyuan Zhang
Ningbo Research Institute, Zhejiang University
College of Information Science and Electronic Engineering, Zhejiang University
NingboTech University
E-mail: cszyzhang@gmail.com

Binh-Son Hua
VinAI, Vietnam
E-mail: binhson.hua@gmail.com

Sai-Kit Yeung
Hong Kong University of Science and Technology
E-mail: saikit@ust.hk

1 Introduction

3D scene understanding is a challenging problem in computer vision. With the wide availability of consumer-grade RGB-D and LiDAR sensors, acquiring 3D scenes has become easier, cheaper, resulting in many mid- and large-scale 3D datasets (Wu et al. 2015a; Chang et al. 2015; Hua et al. 2016; Dai et al. 2017; Armeni et al. 2016; Yi et al. 2016; Uy et al. 2019). One of the popular representations of such 3D data is the 3D point cloud representation. The recent advances of deep learning with 3D point clouds has led to opportunities to revisit and tackle 3D scene understanding from a new perspective.

The basic idea of 3D point clouds deep learning is to let a neural network consume a point cloud directly. A point cloud is a mathematical set and so it fundamentally differs from an image, rendering traditional neural networks unsuitable for 3D point clouds. It is therefore necessary to design a convolution-equivalent operator in the 3D domain that can take a point cloud as input and output its per-point features. In the past few years, significant efforts have been made with promising results along this direction (Qi et al. 2017a,b; Hua et al. 2018; Li et al. 2018; Xu et al. 2018; Zhang et al. 2019b; Zhao et al. 2020).

Despite such research efforts, a property often overlooked in point cloud convolution is rotation invariance. This property arises from the fact that 3D data can

have three degrees of freedom for rotation, making rotation invariance more challenging to achieve. In the 2D domain, a viable solution is to augment the training data with random rotations. However, in 3D, such data augmentation becomes less effective due to the additional degrees of freedom, which can make training prohibitively expensive.

Some previous works have been proposed to learn rotation-invariant features (Zhang et al. 2020, 2019a; Rao et al. 2019; Poulenard et al. 2019; Deng et al. 2018; Chen et al. 2019), which leads to consistent predictions given arbitrarily rotated point clouds. We observe that state-of-the-art methods can improve the feature learning by using local reference frame (LRF) to encode both local and global information (Zhang et al. 2020; Kim et al. 2020b; Thomas 2020). However, LRF usually suffers sign flipping problem in the x and y axes, which makes rotation-invariant convolutions built upon LRF yield features not as distinctive as a translation-invariant convolution does. This can be demonstrated in the result of the object classification task: performing classification with aligned 3D shapes (using translation-invariant convolution) is more accurate than performing the same task with shapes with arbitrary rotations (using rotation-invariant convolution). For example, state-of-the-art methods with rotation invariance (Zhang et al. 2019a; Poulenard et al. 2019; Zhang et al. 2020; Kim et al. 2020b) reported classification accuracies about 86%–89% on ModelNet40 Wu et al. (2015a) while methods without rotation invariance can reach accuracy as high as 93% (Wang et al. 2019; Zhang et al. 2019b). This motivates us to address the limitation of the LRF and design a convolution that gives more informative features to increase the overall performance.

Particularly, we propose an effective and lightweight approach to perform rotation-invariant convolution for point clouds, which is an extension of our previous work (Zhang et al. 2019a, 2020). We propose to make rotation-invariant features more *informative* by local reference axis (LRA), and consider point-point relations, which improves feature distinction as well. Compared to LRF, we show that our LRA is more stable. To the best of our knowledge, our method performs consistent predictions across rotations, and reduces the performance gap compared to translation-invariant convolutions.

In summary, the main contributions of this work are:

- RIConv++, an enhanced version of our previous convolution RIConv (Zhang et al. 2019a). We leverage local reference axis to achieve a stable rotation-invariant representation. We extract *informative* rotation invariant features by considering the relationship between interest points and the neighbors as well as the internal relationship of the neighbors;
- A neural network architecture that stacks RIConv++ for learning rotation-invariant features for 3D point clouds. The network can sense local, semi-global and global context by simply adjusting the neighborhood size and achieves consistent performance across different rotation scenarios;
- Extensive experiments of our method on object classification, object part segmentation and shape retrieval that achieve the state-of-the-art performance under challenging scenarios including an analysis of rotation-invariant features and an ablation study of our neural network.

2 Related Works

3D deep learning began with a focus on regular and structured representations of 3D scenes such as multiple 2D images (Su et al. 2015; Qi et al. 2016; Esteves et al. 2019), 3D volumes (Qi et al. 2016; Li et al. 2016), hierarchical data structures like octree (Riegler et al. 2017) or kd-trees (Klokov and Lempitsky 2017; Wang et al. 2017). Such representations yield good performance, but they face challenges from a practical point of view: they require large memory consumption, have imprecise representation, and are not scalable to high-resolution data. Many recent works in 3D deep learning instead leveraged 3D point cloud, a more compact and intuitive representation compared to volumes and image sets for feature learning.

A daunting task in deep learning with 3D point clouds is how to let a neural network consume a point cloud properly because mathematically, a point cloud is a set, and so to define a valid convolution for a point cloud, it is necessary to ensure that the convolution is invariant to the permutation of the point set. PointNet (Qi et al. 2017a) pioneered the first point cloud convolution with global features by max-pooling per-point features from MLPs. Several follow-up works focus on designing convolutions that can learn local features for a point cloud efficiently (Hua et al. 2018; Qi et al. 2017b; Li et al. 2018; Xu et al. 2018; Wang et al. 2019; Zhang et al. 2019b). Interested readers could refer to the survey by Guo et al. (Guo et al. 2020) for a comprehensive overview of deep learning techniques for 3D point clouds.

However, a missing property in the previously mentioned convolution for point clouds is rotation invariance. To handle rotations, a common approach is to augment the training data with arbitrary rotations, but a disadvantage of this approach is that generalizing the predictions to unseen rotations is challenging, not to mention that the training time becomes longer due to the increased amount of training data. Instead, it is desirable

to design a specific convolution with rotation-invariant features. In the 2D domain, rotation invariance is an appealing property in feature learning where various methods have been proposed such as learning steerable filters (Weiler et al. 2018), performing a log-polar transform of the input (Esteves et al. 2018b) with cylindrical convolutional layers (Kim et al. 2020a), or implementing transformation invariant pooling operators (Laptev et al. 2016). However, these methods are not directly applicable to 3D point clouds due to their difference in both data representation and data dimensionality.

In the 3D domain, rotation invariance has also been specifically built for feature learning of point clouds. (Rao et al. 2019) mapped a point cloud to a spherical domain to define a rotation-invariant convolution. However, the learned features are not purely rotation invariant as the discretized sphere by itself is sensitive to global rotations, resulting in a notable performance drop for objects with arbitrary rotations. To improve the rotation invariant capacity, (Poulenard et al. 2019) proposed to integrate spherical harmonics to a convolution. (Chen et al. 2019) introduced a hierarchical clustering scheme to encode the relative angles between two-point vectors, and vector norm to keeps rotation invariance. (Zhang et al. 2019a) proposed a simple convolution that operates on handcrafted features built from Euclidean distances and angles that are rotation invariant by nature. While consistent results are achieved for arbitrary rotations, only local features are considered which are less descriptive and can cause accuracy degradation. Their follow-up work (Zhang et al. 2020) addressed this limitation by building a global context aware convolution based on anchors and Local Reference Frame (LRF) to achieve rotation invariance. (Kim et al. 2020b) learned rotation invariant local descriptors to aggregate local features based on LRF and applies graph convolutional neural networks. (Thomas 2020) also relied on LRF and used multiple alignment scheme to gain better results. (Li et al. 2021) presented an effective framework to construct both local and global features based on distances, angles and reference points. However, neither LRF nor the global reference are stable enough under noise or outliers, limiting their overall performance.

A great benefit of having rotation invariance property during feature learning is that it allows *consistent* predictions across training/testing scenarios with or without rotations being applied to the data, and they can generalize robustly to inputs with unseen rotations. However, we found that the existing techniques share a common drawback: their performance is inferior to that of a translation-invariant point cloud convolution. This is well reflected in the accuracy of the object classification task on ModelNet40 dataset (Wu et al. 2015a). State-of-the-art translation-invariant convolutions such as PointNet (Qi et al. 2017a), PointNet++ (Qi et al. 2017b), PointCNN (Li et al. 2018), or ShellNet (Zhang et al. 2019b) report 89%–93% of accuracy while techniques with rotation-invariant convolution only report up to 86%–89% of accuracy (Zhang et al. 2019a; Poulenard et al. 2019; Zhang et al. 2020; Kim et al. 2020b; Li et al. 2021). Our work in this paper is dedicated to analyze and reduce this performance gap.

3 Our Method

3.1 Problem Definition

Our goal is to seek a simple but efficient way to perform convolution on a point set such that the translation and rotation invariance property is preserved. Mathematically, let P be the point set; we aim for

$$\mathrm{conv}(\pi(P)) = \mathrm{conv}(P) \tag{1}$$

where $\pi()$ denotes an arbitrary rotation, translation, or point permutation. The traditional convolution is translation invariant by definition. Convolution for a point cloud is specially designed to achieve the permutation invariance property (Qi et al. 2017a). However, techniques that allows point cloud convolution with rotation invariance property has been so far scarce.

An effective solution is to make the input features of P invariant to both translations and rotations, and then design a convolution operator that achieves permutation invariance. In this work, we propose to achieve rotation invariance for point cloud convolution with the state-of-the-art performance by directly leveraging rotation invariant features drawn from low-level geometric cues in the Euclidean space. For completeness, let us first discuss the design of rotation-invariant features in an early version of this work (RIConv (Zhang et al. 2019a)), and then discuss an improved version (RIConv++) with more distinctive features.

3.2 Rotation Invariant Local Features

The design of rotation-invariant features used in RIConv can be explained as in Figure 1. Given a reference point p (red), K nearest neighbors are determined to construct a local point set. The centroid of the point set is denoted as m (blue). We use vector $\overrightarrow{pm}$ as a reference to extract translation and rotation invariant features for all points in the local point set.

Particularly, for a point x in this set, its features are defined as

$$\mathrm{RIF}(x) = [d_0, d_1, \alpha_0, \alpha_1]\,. \tag{2}$$

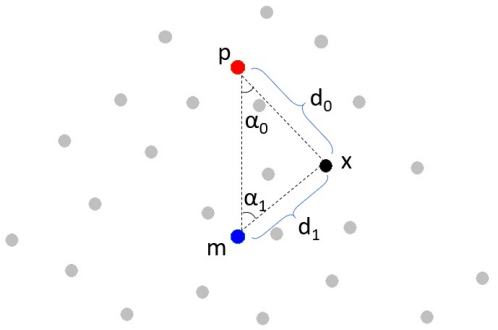

Fig. 1 Rotation invariant features at a point x used by RIConv (Zhang et al. 2019a). The distances and angles are constructed based on the reference vector $\vec{pm}$, where p is the representative point and m the centroid of the local point set. Such features give good performance in the classification task, but are not as distinctive as features learned from translation-invariant convolutions.

Here, d_0 and d_1 represent the distances from x to p and to m, respectively. α_0 and α_1 represent the angles from x towards p and m, as shown in Figure 1. Since such low-level geometric features are invariant under rigid transformations, they are very well suited for our need to make a translation invariant convolution with rotation invariance property. Note that the reference vector $\overrightarrow{pm}$ can also serve as a local orientation indicator and we can use it to build a local coordinate system for convolution. Such rotation invariant features have been used in RIConv (Zhang et al. 2019a) that achieves good classification results on the ModelNet40 dataset (Wu et al. 2015a).

A caveat from the feature extraction scheme above is the stability of the vector $\overrightarrow{pm}$. When the centroid m changes, it can cause $\overrightarrow{pm}$ to be unstable. In this work, we introduce local reference axis (LRA), a more stable reference vector based on the theory of local reference frame (LRF) for rotation-invariant shape descriptors. Similar to vector $\overrightarrow{pm}$, LRA can be used for extracting rotation-invariant features, and for indicating the orientation of the local neighborhood to define the convolution subsequently.

Local Reference Axis Given point p and its neighbors $x_i \in \Omega_p$. A local reference axis (LRA) at p is defined as the eigenvector corresponding to the smallest eigenvalue of the covariance matrix:

$$\Sigma = \sum_{i=1}^{N_{sub}} w_i (x_i - p)(x_i - p)^\top, \tag{3}$$

where N_{sub} is the number of points in the local region and $x_i \in \Omega_p$, and

$$w_i = \frac{m - \|x_i - p\|}{\sum_{i=1}^{N} m - \|x_i - p\|}, \tag{4}$$

where $m = \max_{i=1..N_{sub}}(\|x_i - p\|)$. Intuitively, this weight allows nearby points of p to have large contributions to the covariance matrix, and thus greatly affect the LRA. Points further away from p however can contribute globally to the robustness of the LRA. Such weighted LRA construction is a fundamental step in 3D hand-crafted features (Tombari et al. 2010), which can be easily integrated into our proposed convolution.

Note that the construction of LRA is very similar to that of local coordinate frame (LRF) used in traditional hand-crafted shape descriptor (Tombari et al. 2010). It can be regarded that LRA is the most stable part of LRF (see Section 4.2 for the comparison experiment); in most cases (e.g., locally flat surfaces), LRA is highly similar to the normal vector of the surface. Empirically, we found that features learned with LRA performs as well as those learned with normal vectors. Note that compared to LRF, we do not make use of the two axes tangential to the surface because they could be ambiguous and unstable.

Informative Rotation Invariant Features. Given the definition of LRA, we propose more powerful rotation-invariant features as follows. Given a reference point p, recall that RIConv (Zhang et al. 2019a) only considers the relation between p and its neighbors, measuring Euclidean distances and angles as rotation-invariant features. We propose to additionally consider the distances and angles among the neighbors themselves. An illustration of our proposed features is shown in Figure 2. In this design, the features are kept rotation invariant by definition, and so the framework of RIConv (Zhang et al. 2019a) can work as is. We name such features *Informative* Rotation Invariant Features (IRIF). IRIF transforms each of the neighbor point x_i into a tuple of seven attributes:

$$\text{IRIF}(x_i) = [d, \varphi, \alpha_0, \alpha_1, \alpha_2, \beta_0, \beta_1, \beta_2] \tag{5}$$

where d, α_0, α_1 and α_2 measure the relationship between neighbor point x_i and the reference point p (radial direction):

$$\begin{aligned} d &= \|x_i - p\|, \\ \alpha_0 &= \angle\left(LRA_{x_i}, \overrightarrow{x_i p}\right), \\ \alpha_1 &= \angle\left(LRA_{p}, \overrightarrow{x_i p}\right), \\ \alpha_2 &= S_a \cdot \angle\left(LRA_{x_i}, LRA_{p}\right). \end{aligned} \tag{6}$$

and φ, β_0, β_1, β_2 encode the relationship between x_i and its adjacent neighbor x_{i+1} (clockwise direction):

$$\begin{aligned} \varphi &= \angle\left(\overrightarrow{x_{i+1} p}, \overrightarrow{x_i p}\right), \\ \beta_0 &= \angle\left(LRA_{x_i}, \overrightarrow{x_i x_{i+1}}\right), \\ \beta_1 &= \angle\left(LRA_{x_{i+1}}, \overrightarrow{x_i x_{i+1}}\right), \\ \beta_2 &= S_b \cdot \angle\left(LRA_{x_i}, LRA_{x_{i+1}}\right). \end{aligned} \tag{7}$$

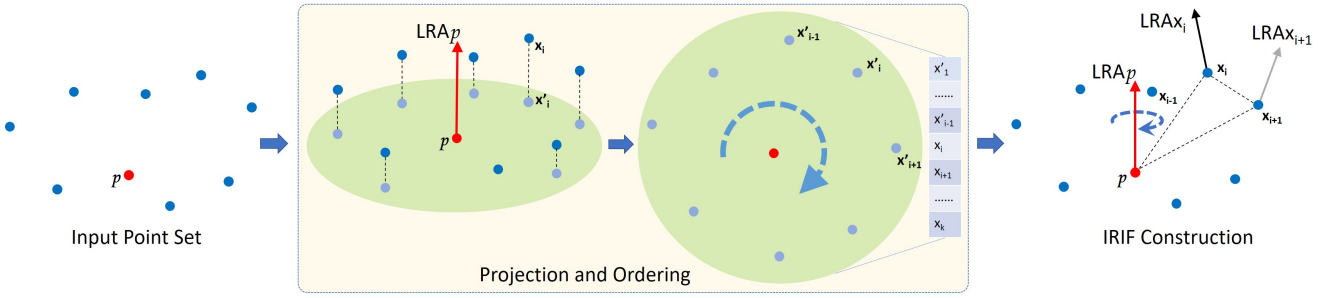

Fig. 2 Informative rotation invariant features (IRIF). The local reference axis (LRA) indicates the local orientation. For a point set with the red point p as reference and grey points as its neighbors (a), clockwise ordering is imposed by projection of the points onto the local tangent disk (b). At each neighbor point, we compute the informative rotation invariant features from the relations between p and its neighbors as well as the relations between the neighbors (c).

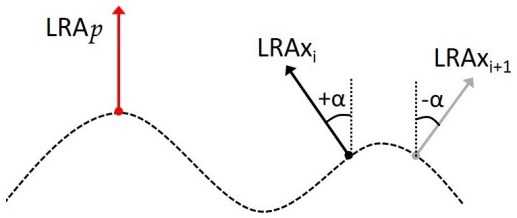

Fig. 3 Illustration of signed angles. Although the absolute angle values of $LRA_{x_{i+1}}$ and LRA_{x_i} relative to LRA_p are the same, their directions are opposite. So the signs are used to indicate the directions.

Here, $\angle$ is computed as the arccos of the normalized vectors. Since arccos returns values in $[0, \pi]$, it has a signed ambiguity as shown Figure 3. Although the absolute angle values are the same, their directions with regard to LRA_p are totally different. To differentiate this, in Equation 5, we propose signed angle to encode both of the angle and direction information between two vectors. We define S_a and S_b to encode the directions as

$$S_a = \begin{cases} +1, & \text{if } \alpha_0 \leq \alpha_1 \\ -1, & \text{otherwise} \end{cases}, \quad S_b = \begin{cases} +1, & \text{if } \beta_0 \leq \beta_1 \\ -1, & \text{otherwise} \end{cases}.$$

Uniqueness of IRIF. During the IRIF construction, it is expected that each point is converted to an unique position in the feature space. The attributes d, α_0, φ in Equation 5 correspond to the radial distance, polar angle, azimuthal angle respectively which uniquely define a 3D point in the local system. Other attributes are used to encode the second order properties making our feature more informative. Note that, for each neighbor point x_i the attribute φ is the angle between its clockwise adjacent neighbor x_{i+1}, and its angle with other neighbor points can be obtained by chain rule. So, each neighbor point has an unique position in the local spherical system theoretically. However, there is a special case that all neighbor points are uniformly distributed along azimuthal direction with exactly same radial distances and same polar angles (e.g., a sphere). In this case, IRIF is no longer unique, but IRIF would still function as a feature with reduced descriptiveness.

Additionally, for the tasks targeting whole objects like classification and retrieval, global uniqueness is also important. This is achieved by enlarging the neighborhood size to include all the interest points. For example, the input of last convolution layer usually contains less number of points with higher dimensions. We can take all the input points into in the last layer to achieve global uniqueness.

3.3 Rotation-Invariant Convolution

From the recipes of informative rotation invariant features and local reference axis, we are now ready to define our convolution. The main steps are detailed in Figure 4.

Particularly, we start by sampling a set of representative points through farthest point sampling strategy which can generate uniformly distributed points. From each of which we perform a set of K-nearest neighbors to obtain local point sets. Let us consider a local point set $\Omega = \{x_i\}$ where x_i represents 3D coordinates of the point i. We define the convolution to learn the features of Ω as

$$\mathbf{f}(\Omega) = \sigma(\mathcal{A}(\{\mathcal{T}(\mathbf{f}_{x_i}) : \forall i\})) \tag{8}$$

This formula indicates that features of each point in the point set are first transformed by $\mathcal{T}$ before being aggregated by the aggregation function $\mathcal{A}$ and passed to an activation function σ. We set the input features to our informative rotation-invariant features $\mathbf{f}_{x_i} = \text{IRIF}(x_i)$. We define the transformation function as

$$\mathcal{T}(\mathbf{f}_{x_i}) = \mathbf{w}_i \cdot \mathbf{f}_{x_i} = \mathbf{f}'_{x_i} \tag{9}$$

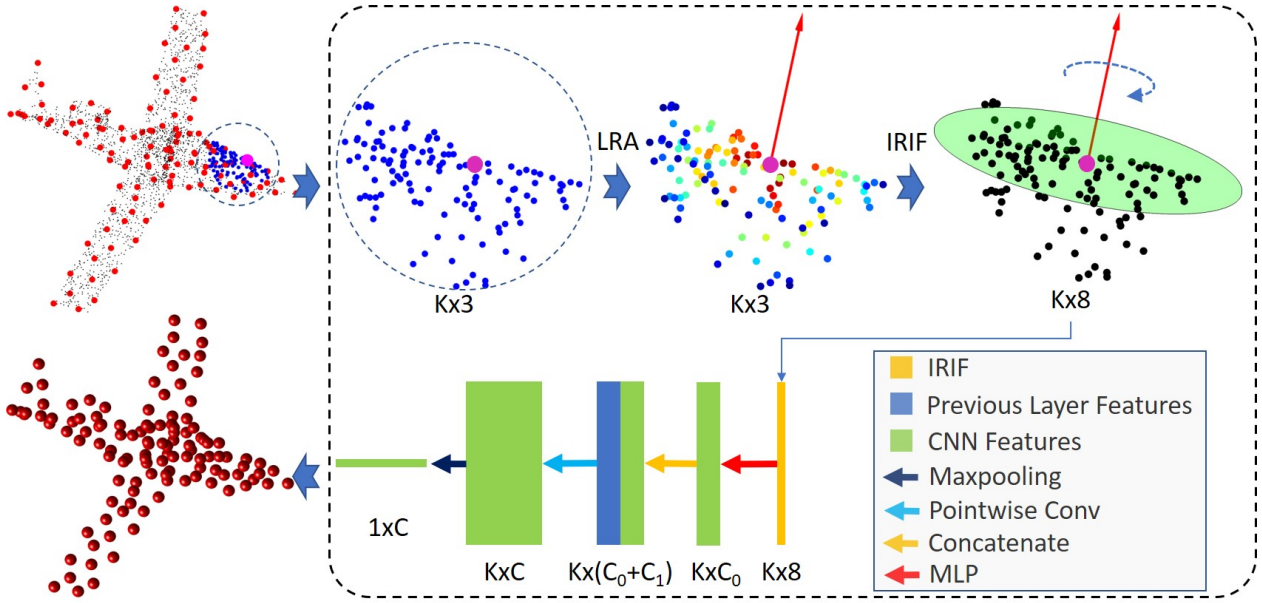

Fig. 4 RIConv++ operator. For an input point cloud, representative points (red dots) are sampled via farthest point sampling. For a reference point p (pink), K nearest neighbors are queried to yield a local point set. Then, we compute the informative rotation invariant features (Section 3.2), which is lifted to a high-dimensional space by a shared multi-layer perceptron (MLP). Concatenated with previous layer features (if any), the features of these local points are further passed to a pointwise convolution, which are finally summarized by maxpooling (Section 3.3). The convolution is applied to all representative points, resulting in output features at such features (denoted as thicker red points).

Algorithm 1 RIConv++ operator.

Input: Reference point p, point set Ω, point features $\mathbf{f}_{prev}$ from previous layer (if any), local reference axes LRA
Output: Convoluted features $\mathbf{f}$

$\mathbf{f} \leftarrow \{\text{IRIF}(x_i) : \forall x_i \in \Omega\}$ * Construct informative rotation invariant features with LRA axes (Section 3.2)
$\mathbf{f} \leftarrow \text{MLP}(\mathbf{f})$; * Lift each feature to a high-dimensional feature
$\mathbf{f}_{in} \leftarrow [\mathbf{f}_{prev}, \mathbf{f}]$ * Concatenate the features from the local and the previous layer (if any)
$\mathbf{f}_{out} \leftarrow \text{conv}(\mathbf{f}_{in})$ * 1D convolution with proper ordering
return $\text{maxpool}(\mathbf{f}_{out})$ * Maxpool features and return

where $\cdot$ indicates the element-wise product, and $\mathbf{w}_i$ is the weight parameter to be learned by the network. Our transformation function is similar to PointNet (Qi et al. 2017a), but applied locally.

A popular choice of the aggregation function $\mathcal{A}$ is maxpooling, which supports permutation invariance in the orders of the input point features (Qi et al. 2017a). Our aggregation function differs in that it includes a 1D convolution kernel and an ordering function to maintain rotation invariance. This has been used in our previous work (Zhang et al. 2019a,b):

$$\mathcal{A}(\{\mathbf{f}'_{x_i}\}) = \text{maxpool}\left(\text{K} \star \text{order}(\{\mathbf{f}'_{x_i}\})\right) \tag{10}$$

where order is a function that sorts the points in a clockwise order based on projecting x_i to the local tangent disk, and K is the 1D convolution kernel. To obtain the ordering, we select one neighbor point as starting point (e.g. x_0), and set $\overrightarrow{x_0 p}$ as reference on the projected disk. Then, we compute the angles between $\overrightarrow{x_i p}$ and $\overrightarrow{x_0 p}$ the ordering is determined by sorting the angles from 0 to 360 degrees. In our implementation, we simply select x_0 by farthest point in the local neighborhood from the reference point p. A benefit of using the farthest point is that the distance between the farthest point and the reference point is less likely to be zero, and so the IRIF features can be validly computed. Note that when the kernel size is 1, point ordering is only used for feature encoding (Equation 7) and not necessary for convolution, and the maxpooling makes the resulting features invariant to point permutation. The detailed steps to perform convolution is shown in the Algorithm 1.

Compared to RIConv (Zhang et al. 2019a) and GCA-Conv (Zhang et al. 2020), our convolution is simpler as it does not require binning. In fact, one of the effects of binning is to fix the instability of the $\overrightarrow{pm}$ vector. In our case, as we find that the LRA is sufficiently stable, we simply apply pointwise convolution as described.

In addition, traditional convolutional neural networks often allow downsampling/upsampling to manipulate the spatial resolution of the input. We build this

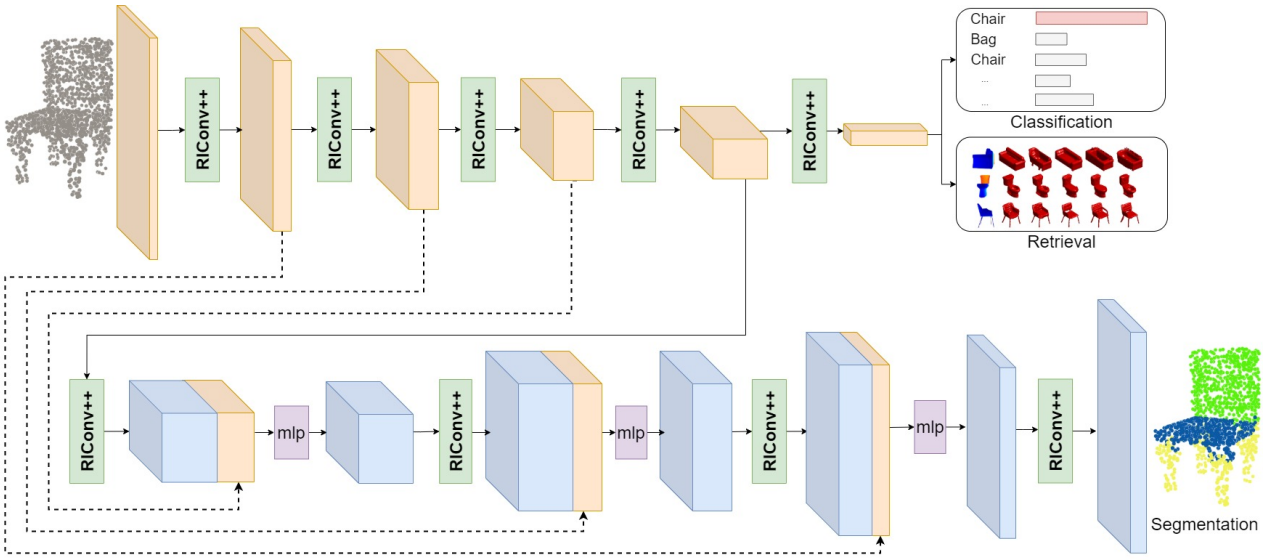

Fig. 5 Our network architecture comprises five convolution layers to extract point cloud features before fully connected layers for object classification and shape retrieval task. We add a decoder with skip connections for segmentation task.

strategy into our convolution by simply treating the sampled point set as the downsampling/upsampling points.

3.4 Network Architecture

We use the proposed convolution to design neural networks for object classification, object part segmentation, semantic segmentation, and shape retrieval, respectively. The architecture is shown in Figure 5 and Table 1. Our classification network has a standard architecture and uses five consecutive layers of convolution (with point downsampling) followed by fully connected layers to output the probability map. As our convolution operator is already designed to handle arbitrary rotation and point orders, we can simply place each convolution one after another. By default, each convolution is followed by a batch normalization and an ReLU activation.

The segmentation network follows an encoder-decoder architecture with skip connections similar to U-Net (Ronneberger et al. 2015). We use an MLP after a skip connection to unify and transform the combined features to have a valid size before applying a deconvolution.

The classification network acts as the encoder, yielding the features in the latent space that can be subsequently decoded into part labels. Unless otherwise mentioned, we use 1024 points for classification/retrieval, 2048 points for part segmentation, and 4096 points for semantic segmentation, respectively.

Our deconvolution is detailed as follows. We define deconvolution in the same way as RIConv++. The difference here is that our convolution outputs to a point subset with more feature channels while deconvolution outputs to a point set with more points compared to the input with fewer feature channels. Particularly, suppose that deconvolution begins with the set of N_l points at layer l. The RIConv++ operator is applied and the features of N_l points are upsampled to a set of N_{l+1} points at the next layer $l+1$. The deconvolution is repeatedly applied until the point cloud reaches the original number of points N. Note that as we have the point subsets during downsampling in the encoder part, we do not need to generate points in upsampling, but just need to reuse the subsets and propagate the features by interpolation.

Neighborhood Size. Our framework is able to integrate both local and global features by simply adjusting the neighborhood size. For classification and retrieval tasks, the global information is more important, so we set the the nearest neighbor size as 8, 16, 32, 64, and 128 respectively for the five layers of convolutions in the encoder to extract features from local to global. For segmentation, we use the same setting for the encoder but carry the features from the 4th layer to the decoder to focus more on the local features and set the neighborhood size as 8, 16, 32, and 32 for the decoder layers.

4 Experiments

We report our evaluation results in this section. We implemented our network in PyTorch, and use a batch size of 16 for all the tasks in training. The optimization is done with an Adam optimizer. The initial learning rate

Table 1 The details of our neural network. Refer to Figure 5 for an illustration of the network architecture, and Algorithm 1 for steps in RIConv++ operator. Here, K is the number of categories, and N is the number of input points.

Module	Output shape
RIConv++	
Input tensor	in_dims × in_points
RIConv++ operator	out_dims × out_points
BatchNorm	out_dims × out_points
ReLU	out_dims × out_points
Classification / Retrieval	
Input tensor	$3 \times N$
RIConv++	32×1024
RIConv++	64×512
RIConv++	128×256
RIConv++	256×128
RIConv++	512×1
Fully connected	512×1
Fully connected	256×1
Softmax	$K \times 1$
Segmentation	
Input tensor	$3 \times N$
RIConv++	64×512
RIConv++	128×256
RIConv++	256×128
RIConv++	512×64
RIConv++	512×128
Skip connection	768×128
MLP	512×128
RIConv++	512×256
Skip connection	640×256
MLP	256×256
RIConv++	256×512
Skip connection	320×512
MLP	128×512
RIConv++	$K \times N$

is set to 0.001. Our training is executed on a computer with an Intel(R) Core(TM) i7-10700K CPU equipped with a NVIDIA GTX 2080ti GPU.

We evaluate our method with object classification, shape retrieval, object part segmentation, and semantic segmentation. For object classification and retrieval, we train for 200 epochs, and the network usually converges within 150 epochs. For object part segmentation, we train for 200 epochs, and the network usually converges within 150 epochs. For semantic segmentation, we train for 40 epochs. It takes about 2 hours for the training to converge for classification, about 15 hours for part segmentation, and about 40 hours for semantic segmentation.

Following Esteves et al. (2018a), we perform experiments in three cases: (1) training and testing with data augmented with rotation about gravity axis (z/z), (2) training and testing with data augmented with arbitrary SO3 rotations (SO3/SO3), and (3) training with data by z-rotations and testing with data by SO3 rotations (z/SO3). The first case is commonly used for evaluating translation-invariant point cloud learning methods, and the last two cases are for evaluating rotation invariance. In general, convolution with rotation invariance is expected to work well for case (3) even though the network is not trained with data augmented with SO3 rotations.

In general, our result demonstrates the effectiveness of the rotation invariant convolution we proposed. Our networks yield very *consistent* results despite that our networks are trained with a limited set of rotated point clouds and tested with arbitrary rotations. To the best of our knowledge, there is no previous work for point cloud learning that can achieve the same level of accuracy with the same level of consistency despite that some methods (Esteves et al. 2018a; Rao et al. 2019) demonstrated good performance when trained with a particular set of rotations. We detail our evaluations below.

4.1 Object Classification

The classification task is trained on the ModelNet40 variant of the ModelNet dataset (Wu et al. 2015b). ModelNet40 contains CAD models from 40 categories such as airplane, car, bottle, dresser, etc. By following Qi et al. (2017b), we use the preprocessed 9,843 models for training and 2,468 models for testing. The input point cloud size is 1024, with each point has the attributes (x, y, z, nx, ny, nz) which are 3D coordinates and 3D normals in the Euclidean space.

We use the encoder layers of Figure 5 which outputs one feature vector to train the classifier. Particularly, our network outputs one feature vector of length 512 to the classifier, which is then passed through an MLP implemented by fully connected layers, resulting in 128×40 category predictions.

We use two criteria for evaluation: accuracy and accuracy standard deviation. Accuracy is a common metric to measure the performance of the classification task. In addition, accuracy deviation measures the consistency of the accuracy scores in three tested cases. In general, it is expected that methods that are rotation invariant should be insusceptible to the rotation used in the training and testing data and therefore has a low deviation in accuracy.

The evaluation results are shown in Table 2. As can be seen, our method achieves the state-of-the-art performance in all cases. More importantly, our method has almost zero accuracy deviation. Non-rotation invariant point cloud learning methods exhibit large accuracy deviations especially in the extreme z/SO3 case. This case

Table 2 Comparisons of the classification accuracy (%) on the ModelNet40 dataset. On average, our method has the best accuracy and lowest accuracy deviation in all cases including with and without normals. Here, 'nor' means 'normal'.

	Method	Format	Input Size	Params.	z/z↑	SO3/SO3↑	z/SO3↑	Std.↓
Traditional	VoxNet (Maturana and Scherer 2015)	voxel	30^3	0.90M	83.0	87.3	-	3.0
	SubVolSup (Qi et al. 2016)	voxel	30^3	17.00M	88.5	82.7	36.6	28.4
	MVCNN 80x (Su et al. 2015)	view	80×224^2	99.00M	90.2	86.0	81.5	4.3
	PointNet (Qi et al. 2017a)	xyz	1024×3	3.50M	87.0	80.3	21.6	41.0
	PointCNN (Li et al. 2018)	xyz	1024×3	0.60M	91.3	84.5	41.2	27.2
	PointNet++ (Qi et al. 2017b)	xyz + nor	1024×6	1.40M	89.3	85.0	28.6	33.8
	DGCNN (Wang et al. 2019)	xyz	1024×3	1.84M	92.2	81.1	20.6	38.5
	RS-CNN (Liu et al. 2019)	xyz	1024×3	1.41M	90.3	82.6	48.7	22.1
Rotation-invariant	Spherical CNN (Esteves et al. 2018a)	voxel	2×64^2	0.50M	88.9	86.9	78.6	5.5
	RIConv (Zhang et al. 2019a)	xyz	1024×3	0.70M	86.5	86.4	86.4	0.1
	SPHNet (Poulenard et al. 2019)	xyz	1024×3	2.90M	87.0	87.6	86.6	0.5
	SFCNN (Rao et al. 2019)	xyz	1024×3	-	**91.4**	90.1	84.8	3.5
	ClusterNet (Chen et al. 2019)	xyz	1024×3	1.40M	87.1	87.1	87.1	**0.0**
	GCAConv (Zhang et al. 2020)	xyz	1024×3	0.41M	89.0	89.2	89.1	**0.0**
	RI-GCN (Kim et al. 2020b)	xyz	1024×3	4.38M	89.5	89.5	89.5	**0.0**
	RIF (Li et al. 2021)	xyz	1024×3	-	89.4	89.3	89.4	**0.0**
	Ours	xyz	1024×3	**0.42M**	91.2	**91.2**	**91.2**	**0.0**
	RI-GCN (Kim et al. 2020b)	xyz + nor	1024×6	4.38M	91.0	91.0	91.0	**0.0**
	Ours	xyz + nor	1024×6	**0.42M**	**91.3**	**91.3**	**91.3**	**0.0**

is exceptionally hard for methods that rely on data augmentation to handle rotations (Qi et al. 2017a,b). In our observation, such techniques are only able to generalize within the type of rotation they are trained with, and generally fail in the z/SO3 test. This applies to both voxel-based and point-based learning techniques. By contrast, our method has almost no performance difference in three test cases, which confirms the robustness of the rotation invariant geometric cues in our convolution. The success of our method is attributed to two factors: the informative rotation-invariant features (Section 3.2), and the use of large neighborhood in the last layer of the network that naturally widens the perceptive field and captures global features.

Table 2 also includes a comparison on the use of normal vectors for feature learning as follows. For our method with input format xyz, we use the LRA as the reference axis, while with input format $xyz + nor$, we use normal vectors as the reference axis. This is different from PointNet++ and RI-GCN which treat normals as extra features. In both cases, our method has almost similar performance difference (0.1% accuracy difference) which means that our LRA is as descriptive as normal vectors. Our method also outperforms both PointNet++ and RI-GCN in both cases. Note that we achieve this performance without voting, a test-time augmentation scheme to boost the classification result used by RI-GCN. When voting is disabled, RI-GCN has 1% accuracy drop. Voting also slows down the inference in real applications.

Network Parameters. The capability to handle rotation invariance also has a great effect on the number of network parameters. For networks that rely on data augmentation to handle rotations, it requires more parameters to 'memorize' the rotations. Networks that are designed to be rotation invariant, such as spherical CNN (Esteves et al. 2018a) and ours, have very compact representations. In terms of the number of trainable parameters, our network has 0.4 millions (0.4M) of trainable parameters, which is the most compact network in our evaluations. Among the tested methods, only spherical CNN (Esteves et al. 2018a) (0.5M) and PointCNN (0.6M) have similar compactness. Our network has 9× less parameters than PointNet (3.5M), more than 3× less than PointNet++ (1.4M). The good balance between trainable parameters, accuracy and accuracy deviations makes our method more robust for practical use.

4.2 An Analysis of Rotation-Invariant Features

We compare the performance with RIConv (Zhang et al. 2019a) and GCAConv (Zhang et al. 2020) using same number of layers and same number of representative points in each layer. Here, we set the total number of layers as 3 with 512, 128, 32 representative points and 64, 32, 16 nearest neighbors, respectively. The results are shown in Table 3. In general, we observe that our informative rotation invariant features (IRIF) is more accurate than original features used in RIConv (Zhang et al. 2019a) and GCAConv (Zhang et al. 2020).

Table 3 Performance comparisons with our previous work, RIConv (Zhang et al. 2019a) and GCAConv (Zhang et al. 2020) (%) on the ModelNet40 dataset with the same representative points and same number of layers. It shows the effectiveness of our proposed features.

Method	Core Features	OA
RIConv (Zhang et al. 2019a)	RIF	86.5
GCAConv (Zhang et al. 2020)	LRF Transform	89.0
Ours (w/o normal)	IRIF	**89.8**
Ours (w/ normal)	IRIF	**90.3**

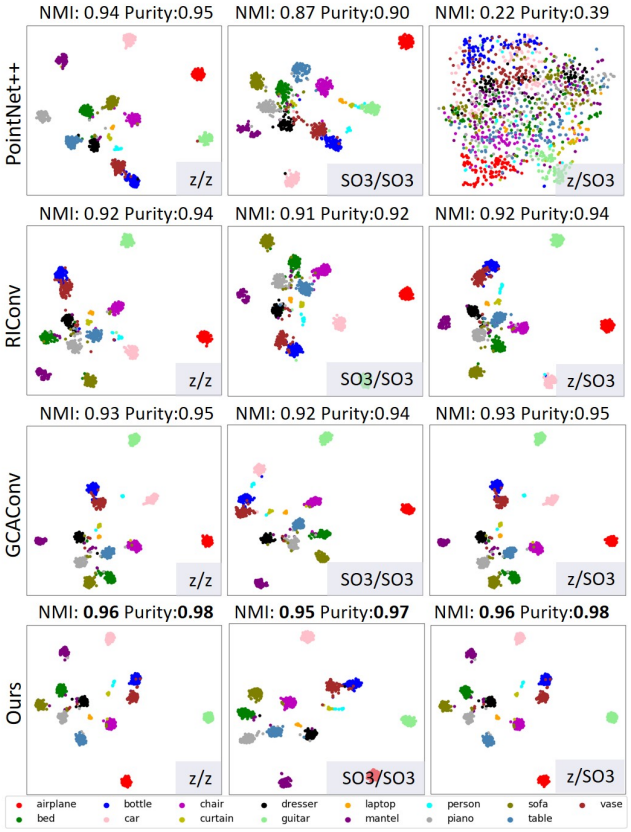

Fig. 6 t-SNE comparisons of the latent features for PointNet++ (Qi et al. 2017b), RIConv (Zhang et al. 2019a), GCAConv (Zhang et al. 2020) and our method under three different rotation settings. The clusters in the t-SNEs show that to make good decisions in object classification, it is desirable to have the cluster boundaries as separated as possible.

We further visualize the latent space learned by the neural networks using t-SNE (van der Maaten and Hinton 2008). The results are shown in Figure 6. We measure the clustering quality by two metrics, the normalized mutual information (NMI) and purity (Manning et al. 2008). We demonstrate three scenarios for object classification: z/z, SO3/SO3, and z/SO3. As can be seen, latent space learned by rotation-invariant convolution such as RIConv (Zhang et al. 2019a) does not exhibit good discrimination among classes. The main difference

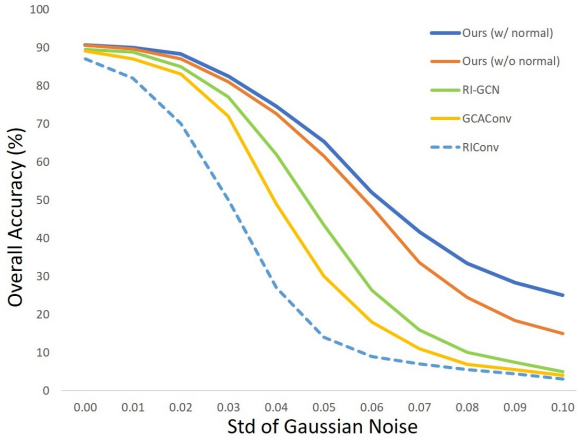

Fig. 7 Performance of our method in the presence of additive Gaussian noise on the ModelNet40 dataset. Our method outperforms all existing rotation-invariant convolutions.

between such convolution and traditional point cloud convolution is that it no longer works with point coordinates at start. In the case of RIConv, the points are transformed into Euclidean-based features including distances and angles, which are not as unique as point coordinates since many points can share the same distance and angles. This is well reflected in the t-SNE in the first column (z/z) in Figure 6. PointNet++ (Qi et al. 2017b) has a good separation among the clusters while RIConv (Zhang et al. 2019a) has more condensed clusters in the center, resulting in more ambiguities during classification. The clusters by GCAConv (Zhang et al. 2020) and our method are more similar to PointNet++, which explains their similar performance.

Similarly, in the second column (SO3/SO3), all methods have similar clustering, which explains their similar performance in the classification (see more quantitative comparisons in Table 2). Finally, the third column (z/SO3) highlights the strength of rotation-invariant convolutions as they can still maintain consistent predictions and generalize well to unseen conditions. In this case, the t-SNEs show that PointNet++ cannot generalize effectively.

4.3 Robustness to Noise

In real applications, the point cloud data produced from scanners usually contains noise. Here we conduct an experiment to verify the robustness of existing rotation-invariant convolutions to noise. We sample and add noise from zero-mean Gaussian distributions $N(0, \sigma^2)$ to the data, and compare the classification performance of our method with RIConv (Zhang et al. 2019a), GCAConv (Zhang et al. 2020) and RI-GCN (Kim et al. 2020b). The noise level can be controlled by adjusting σ^2.

In Figure 7, we plot the accuracy against different noise level σ. It can be seen that our method outperforms all remaining methods. In RIConv, the use of vector $\overrightarrow{pm}$ (from the representative point to the local centroid) as reference axis is unstable under noise. GCAConv and RI-GCN uses LRF which is also vulnerable to noise.

4.4 LRA/LRF Comparison

As the LRA is the basis to the define the Informative Rotation Invariant Features (IRIF), it is necessary to analyse the stability of LRA separately. A popular way to evaluate the robustness of LRA or LRF is to benchmark the repeatability. We follow Guo et al. (2013) (see their section 3.3) to conduct this experiment. Noted that there are also methods that solve LRFs for mesh such as MeshHog (Zaharescu et al. 2009) and RoPS (Guo et al. 2013). In this study we assume no normal vectors or triangle faces so we omit such methods in our comparison. We use six models from the Stanford 3D Scanning Repository (Curless and Levoy 1996) (Figure 8 top). The scenes are created by resampling the models down to 1/2 of their original mesh resolution with Gaussian noise added (0.5 mesh resolution).

From each model, 1000 points are randomly selected and their correspondences in the scene are obtained by searching the closest point in the Euclidean space. Let's denote the pair of points as (p_{si}, p_{mi}) from scene and model respectively. The LRFs or LRAs for these two points are computed as V_{s_i} and V_{m_i}. To measure the similarity between them, we use the error evaluation metric provided by Mian et al. (2010):

$$e_i = \arccos\left(\frac{\mathrm{Tr}(V_{s_i}V_{m_i}) - 1}{2}\right)\frac{180}{\pi}. \tag{11}$$

Ideally, e_i is zero when there is no error. We compare with LRF used in SHOT (Tombari et al. 2010) and pm vector used in RIConv (Zhang et al. 2019a). The results are shown in Figure 8, where the horizontal axis indicates the angular error range and the vertical axis represents the percentage of points. The more points fall into left lower error range, the better of the methods. As can be seen, LRA have much more low-range angular errors than other methods, and has significantly less high-range errors.

4.5 Ablation Studies

IRIF Design. We first experiment by turning on/off different rotation invariant components used in the IRIF construction (3.2). The result of this experiment is shown in Table 4. Model A is our baseline setting with all rotation invariant features activated. Model B has only angle features, and Model C has only distance features. From the comparison, we can see that turning off either feature types can deteriorate the results. When only distances are employed (Model C), the accuracy decreases to 83.3%. In Model D, we only keep features on the radial direction with d, α_0, α_1 and α_2 such that

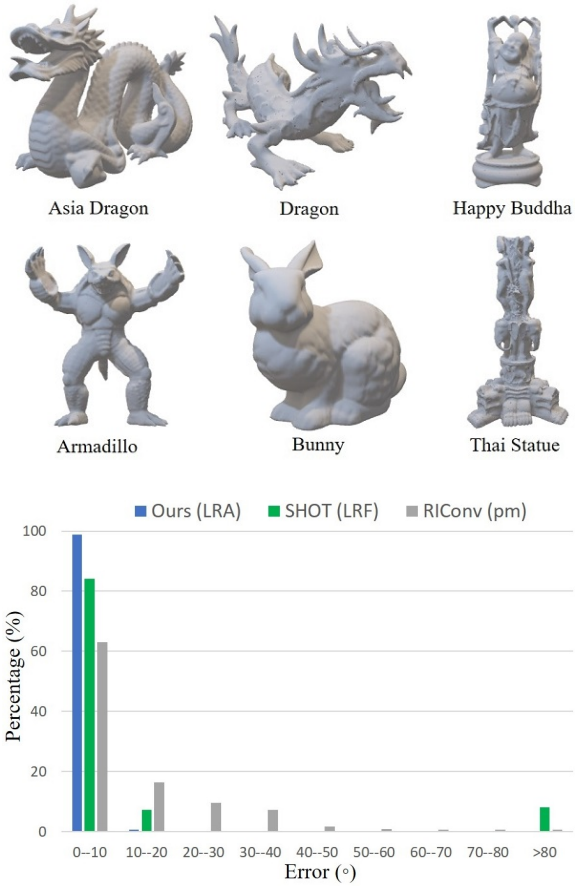

Fig. 8 Repeatability of reference axis. We plot the histogram of errors of our LRA, LRF (Tombari et al. 2010), and the pm vector in RIConv (Zhang et al. 2019a) for six models from the Stanford 3D Scanning Repository Curless and Levoy (1996). LRA has the best lower range errors.

Table 4 An evaluation of our features design. By combining distances and angles, relations between the representative points and the neighbors, relations among the neighbors, Model A with IRIF can achieve performance on ModelNet40 similar to PointCNN (Li et al. 2018) using xyz coordinates.

Model	φ	β_0	β_1	β_2	d	α_0	α_1	α_2	Acc.
A	✓	✓	✓	✓	✓	✓	✓	✓	91.2
B		✓	✓	✓		✓	✓	✓	90.5
C	✓				✓				83.3
D					✓	✓	✓	✓	88.4

Table 5 Performance comparisons with different kernel sizes. Kernel size 1 has the best speed-accuracy trade-off.

Kernel Size	Time / Epoch	Params.	Acc.
1	40s	0.40M	91.2
3	63s	0.83M	91.2
5	73s	1.29M	91.0
7	85s	1.73M	91.0

Table 6 Application of IRIF features on different network architecture.

Method	z/z	SO3/SO3	z/SO3	Acc. Std
PN++	89.3	85.0	28.6	33.8
PN++ (IRIF)	89.1	89.1	89.0	**0.0**
DGCNN	92.2	81.1	20.6	38.5
DGCNN (IRIF)	90.2	90.1	90.2	**0.0**

only the relations between the reference point and its neighbors are considered. This setting is used in RIConv (Zhang et al. 2019a). Compared to Model A, it shows that our proposed features is more effective than those by RIConv, and the improvement is explained by the additional consideration of the relations among the neighbor points.

Kernel Size. For the pointwise convolution in Algorithm 1, different kernel sizes can be chosen. We compare the performance with different kernel sizes as shown in Table 5. With larger kernels, the running time and the number of parameters also increases. We observed that larger kernels (5 and 7) do not correspond to better performance because our rotation-invariant features only involves a point x_i and its immediate neighbors, which makes a kernel up to size 3 sufficient to capture the local features. In our experiments, we choose kernel size 1 for the best speed and accuracy trade-off.

IRIF on different network architectures IRIF can be applied for other architectures to realize rotation invariance. Here, we experiment with PointNet++ (Qi et al. 2017b) and DGCNN (Wang et al. 2019) by replacing their xyz coordinates input with IRIF. The results are shown in Table 6. We can see that rotation invariance can be achieved.

Training Statistics. We further analyze the training convergence and measure the accuracy on the test set on different training epochs. We use the z/SO3 setting. The plot is shown in Figure 9. It can be seen that RIConv (Zhang et al. 2019a) converges to a flat accuracy curve after about 20 epochs. Our new method converges with higher accuracy, and the accuracy still increases by

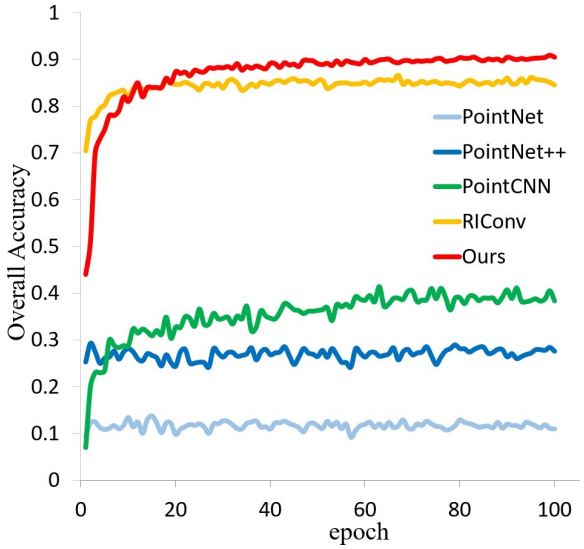

Fig. 9 Overall accuracy versus training epochs plot of object classification under z/SO3 mode on the ModelNet40 dataset.

100 epochs. Both methods outperform standard point cloud methods (Qi et al. 2017a,b; Li et al. 2018) by a large margin as expected for the z/SO3 setting.

4.6 Real World 3D Point Cloud Classification

Real-world point cloud data usually contains missing data, occlusions, and non-uniform density. We evaluate the classification performance on ScanObjectNN (Uy et al. 2019), a real-world dataset of 3D point clouds captured by RGB-D camera. The data comprises 2902 objects classified into 15 categories sampled from real-world indoor scenes. For our evaluation, we use the processed files and choose the easiest variant OBJ_ONLY and OBJ_BG (only object without/with background points, without rotation, translation, and scaling) and the hardest variant PB_T50_RS (with 50% bounding box translation, rotation around the gravity axis, and random scaling that result in rotated and partial data). The results are shown in Table 7. It can be seen that our method outperforms other existing approaches across all the scenarios and only slightly worse than PointCNN under the z/z scenario. This verifies that RIConv++ is also effective on real-world datasets. Note that we only test the 'w/o normal' case as the normal vectors are not provided in the processed files of this dataset.

The experiment with ScanObjectNN also demonstrates the robustness of our method in the presence of occlusion or background points. Firstly, we can see that our method works well for objects in ScanObjectNN despite that they are incomplete scans due to occlusions, which demonstrates the effectiveness of our rotation-invariant features (IRIF) and the construction of local

Table 7 Comparisons of real world 3D point cloud classification on ScanObjectNN (Uy et al. 2019) dataset. We tested on the OBJ_ONLY (object without background), OBJ_BG (object with background) and PB_T50_RS (hardest) variant.

Method	OBJ_ONLY			OBJ_BG			PB_T50_RS		
	z/z	SO3/SO3	z/SO3	z/z	SO3/SO3	z/SO3	z/z	SO3/SO3	z/SO3
PointNet (Qi et al. 2017a)	79.2	57.5	28.7	73.3	54.7	16.7	68.2	42.2	17.1
PointNet++ (Qi et al. 2017b)	84.3	57.1	25.6	82.3	47.4	15.0	77.9	60.1	15.8
PointCNN (Li et al. 2018)	85.5	66.9	21.9	**86.1**	63.7	14.6	78.5	51.8	14.9
DGCNN (Wang et al. 2019)	86.2	73.6	20.7	82.8	71.8	17.7	78.1	63.4	16.1
RIConv (Zhang et al. 2019a)	79.8	79.8	79.8	78.4	78.2	78.4	68.1	68.3	68.3
GCAConv (Zhang et al. 2020)	80.1	80.3	80.1	78.2	78.1	78.2	69.8	70.0	69.8
Ours (w/o normal)	**86.2**	**86.2**	**86.2**	85.6	**85.6**	**85.6**	**80.3**	**80.3**	**80.3**

Table 8 Comparisons of 3D shape retrieval on the ShapeNet Core (Wu et al. 2015b). The accuracy (%) is reported based on the standard evaluation metrics including precision, recall, f-score, mean average precision (mAP) and normalized discounted cumulative gain (NDCG).

Method	micro					macro					
	PN	R@N	F1@N	mAP	NDCG	PN	R@N	F1@N	mAP	NDCG	Score
Furuya and Ohbuchi (2016)	81.4	68.3	70.6	65.6	75.4	60.7	53.9	50.3	47.6	56.0	56.6
Tatsuma and Aono (2009)	70.5	76.9	71.9	69.6	78.3	42.4	56.3	43.4	41.8	47.9	55.7
Bai et al. (2016)	66.0	65.0	64.3	56.7	70.1	44.3	50.8	43.7	40.6	51.3	48.7
Esteves et al. (2018a)	71.7	73.7	-	68.5	-	45.0	55.0	-	44.4	-	56.5
SFCNN (Rao et al. 2019)	77.8	75.1	75.2	70.5	81.3	65.6	53.9	53.6	48.3	58.0	59.4
GCAConv (Zhang et al. 2020)	82.9	76.3	74.8	70.8	81.3	66.8	55.9	51.2	49.0	58.2	**61.2**
RIF (Li et al. 2021)	82.1	73.7	74.1	70.7	80.5	51.2	**66.4**	**55.8**	**51.0**	56.0	60.9
Ours	**83.2**	**77.2**	**75.2**	**71.3**	**81.8**	**68.1**	58.2	53.7	50.2	**58.8**	60.7

reference axes (LRA). Secondly, when additional background points are added to the objects as demonstrated in the OBJ_BG variant, comparing between OBJ_ONLY and OBJ_BG, we observe that our method is more tolerant and has less accuracy drop (-0.6%) compared to previous methods (RIConv with -1.4% drop and GCAConv with -1.9% drop in z/z case).

4.7 Shape Retrieval

Another popular task to evaluate rotation invariance on 3D shape is shape retrieval (Savva et al. 2016). Here we conducted experiments on ShapeNet Core (Wu et al. 2015b), following the perturbed protocol of the SHREC'17 3D shape retrieval contest (Savva et al. 2016) and the experiment setting of SFCNN (Rao et al. 2019). We use the same output features from the bottleneck layer in the network (similar to features used in the classification task; see Figure 5). We compare our method with methods proposed in SHREC'17 (Furuya and Ohbuchi 2016; Tatsuma and Aono 2009; Bai et al. 2016) and two recent methods on rotation-invariant convolution (Esteves et al. 2018a; Rao et al. 2019). The results are shown in Table 8. Similar to the classification task, our method achieves the state-of-the-art result, outperforming previous methods for most evaluation metrics.

4.8 Object Part Segmentation on ShapeNet

We also evaluated our method with the object part segmentation where each point of the input point cloud is predicted with a part label. We train and test with the ShapeNet dataset (Chang et al. 2015) that contains $16,880$ CAD models in 16 categories. Each model is annotated with 2 to 6 parts, resulting in a total of 50 object parts. We follow the standard train/test split with $14,006$ models for training and $2,874$ models for testing, respectively.

The evaluation results are shown in Table 9. Our method outperforms translation-invariant convolution methods significantly in the SO3/SO3 and z/SO3 scenario. Compared to previous rotation-invariant convolution methods, our method also has better performance. This result aligns well with the performance reported in the object classification task. Our method also has consistent performance for both rotation cases, which empirically confirms the rotation invariance in our convolution. We illustrate the qualitative results by error maps as shown in Figure 10. By comparing with the

Table 9 Comparisons of object part segmentation performed on ShapeNet dataset (Chang et al. 2015). The mean per-class IoU (mIoU, %) is used to measure the accuracy under two challenging rotation modes: SO3/SO3 and z/SO3.

Method	SO3/SO3	z/SO3
PointNet (Qi et al. 2017a)	74.4	37.8
PointCNN (Li et al. 2018)	71.4	34.7
DGCNN (Wang et al. 2019)	73.3	37.4
RS-CNN (Liu et al. 2019)	72.5	36.5
RIConv (Zhang et al. 2019a)	75.5	75.3
GCAConv (Zhang et al. 2020)	77.3	77.2
RI-GCN (Kim et al. 2020b)	77.0	77.0
RIF (Li et al. 2021)	79.4	79.2
Ours (w/o normal)	**80.3**	**80.3**
PointNet++ (Qi et al. 2017b)	76.7	48.2
SpiderCNN (Xu et al. 2018)	72.3	42.9
Ours (w/ normal)	**80.5**	**80.5**

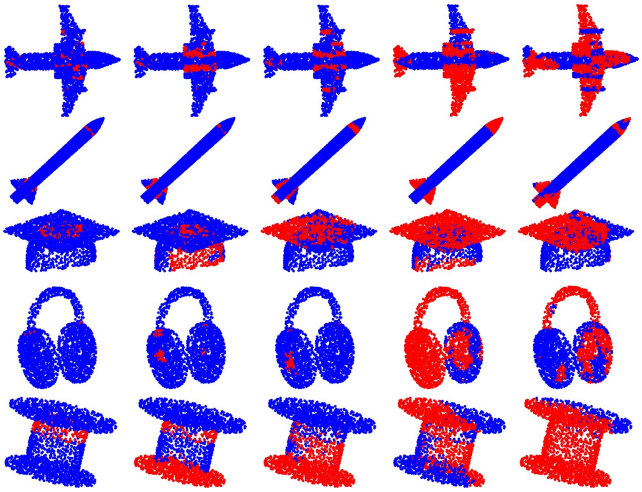

Fig. 10 Comparison of part segmentation error maps on ShapeNet dataset under z/SO3 mode. The red points indicate wrong segmentations.

groundtruth, we plot the wrong segmentation points as red. The plot clearly shows that our predictions are the closest to the ground truth.

4.9 Large-scale scene segmentation on S3DIS

We conduct an experiment for the scene segmentation on the S3DIS dataset (Armeni et al. 2016) which is a large-scale indoor scene dataset comprising 3D scans from Matterport scanners in 6 indoor areas including 271 rooms with each point is annotated with one of the semantic labels from 13 categories. In this experiment, we use Area-5 for testing to better measure the generalization ability, and report the results in SO3/SO3 and z/SO3 scenario. The results are shown in Table 10. From the results, we can see that RIConv++ outperforms RIConv (Zhang et al. 2019a) and GCAConv (Zhang et al. 2020), and works consistently for both scenarios.

Table 10 Comparisons of the semantic segmentation accuracy (mIoU, %) on the S3DIS dataset (Area-5).

Method	SO3/SO3	z/SO3
PointNet (Qi et al. 2017a)	35.2	20.8
PointCNN (Li et al. 2018)	43.5	23.6
RIConv (Zhang et al. 2019a)	53.3	53.2
GCAConv (Zhang et al. 2020)	55.8	55.7
Ours (w/o normal)	**57.0**	**57.1**

5 Conclusion

In this work, we revisited the design of rotation-invariant features for 3D point cloud convolution, and proposed RIConv++ for 3D point cloud convolution that satisfies rotation invariance property. The effectiveness of RIConv++ comes from the powerful and stable rotation invariant features and the flexible neighborhood size employed in the convolution. We used local reference axis (LRA) instead of the commonly used local reference frame (LRF) to construct stable rotation invariant features. The relationship between the representative point and its neighbors and relationship among neighbors are used in tandem to make our convolution much informative. The local-global information can be captured by simply using different neighborhood size in each convolution. Such a design achieves the state-of-the-art results in various tasks such as object classification, shape retrieval, and object part segmentation.

As our newly proposed convolution can closely match the performance of state-of-the-art translation-invariant convolutions, our work opens up opportunities to further reduce the performance gap between rotation- and translation-invariant convolution especially in the presence of noise. Our method also shows that handcrafted rotation-invariant features, when used properly, can lead to compelling results in 3D deep learning. There remains an open question: it would be of great interest to design a convolution that can learn rotation-invariant features automatically, without the need of handcrafted features, which could allow the applications of rotation-invariant features to more data domains.

Acknowledgements We thank the anonymous reviewers for their constructive comments. This research project is supported by the grant from Ningbo Research Institute of Zhejiang University (1149957B20210125), and partially supported by an internal grant from HKUST (R9429).

References

Armeni I, Sener O, Zamir AR, Jiang H, Brilakis I, Fischer M, Savarese S (2016) 3d semantic parsing of large-scale indoor spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1534–1543 1, 14

Bai S, Bai X, Zhou Z, Zhang Z, Jan Latecki L (2016) Gift: A real-time and scalable 3d shape search engine. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5023–5032 13

Chang AX, Funkhouser TA, Guibas LJ, Hanrahan P, Huang QX, Li Z, Savarese S, Savva M, Song S, Su H, Xiao J, Yi L, Yu F (2015) Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:151203012 1, 13, 14

Chen C, Li G, Xu R, Chen T, Wang M, Lin L (2019) Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4994–5002 2, 3, 9

Curless B, Levoy M (1996) A volumetric method for building complex models from range images. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp 303–312 11

Dai A, Chang AX, Savva M, Halber M, Funkhouser T, Niessner M (2017) Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 5828–5839 1

Deng H, Birdal T, Ilic S (2018) Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In: Proceedings of the European Conference on Computer Vision, pp 602–618 2

Esteves C, Allen-Blanchette C, Makadia A, Daniilidis K (2018a) Learning so (3) equivariant representations with spherical cnns. In: Proceedings of the European Conference on Computer Vision (ECCV), pp 52–68 8, 9, 13

Esteves C, Allen-Blanchette C, Zhou X, Daniilidis K (2018b) Polar transformer networks. In: International Conference on Learning Representations 3

Esteves C, Xu Y, Allen-Blanchette C, Daniilidis K (2019) Equivariant multi-view networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1568–1577 2

Furuya T, Ohbuchi R (2016) Deep aggregation of local 3d geometric features for 3d model retrieval. In: BMVC, vol 7, p 8 13

Guo Y, Sohel F, Bennamoun M, Lu M, Wan J (2013) Rotational projection statistics for 3d local surface description and object recognition. International journal of computer vision 105(1):63–86 11

Guo Y, Wang H, Hu Q, Liu H, Liu L, Bennamoun M (2020) Deep learning for 3d point clouds: A survey. IEEE transactions on pattern analysis and machine intelligence 43(12):4338–4364 2

Hua BS, Pham QH, Nguyen DT, Tran MK, Yu LF, Yeung SK (2016) Scenenn: A scene meshes dataset with annotations. In: 2016 Fourth International Conference on 3D Vision, pp 92–101 1

Hua BS, Tran MK, Yeung SK (2018) Pointwise convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 984–993 1, 2

Kim J, Jung W, Kim H, Lee J (2020a) Cycnn: a rotation invariant cnn using polar mapping and cylindrical convolution layers. arXiv preprint arXiv:200710588 3

Kim S, Park J, Han B (2020b) Rotation-invariant local-to-global representation learning for 3d point cloud. Advances in Neural Information Processing Systems 33:8174–8185 2, 3, 9, 10, 14

Klokov R, Lempitsky V (2017) Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In: Proceedings of the IEEE International Conference on Computer Vision, pp 863–872 2

Laptev D, Savinov N, Buhmann JM, Pollefeys M (2016) Ti-pooling: transformation-invariant pooling for feature learning in convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 289–297 3

Li X, Li R, Chen G, Fu CW, Cohen-Or D, Heng PA (2021) A rotation-invariant framework for deep point cloud analysis. IEEE Transactions on Visualization and Computer Graphics 3, 9, 13, 14

Li Y, Pirk S, Su H, Qi CR, Guibas LJ (2016) Fpnn: Field probing neural networks for 3d data. In: Advances in Neural Information Processing Systems, pp 307–315 2

Li Y, Bu R, Sun M, Wu W, Di X, Chen B (2018) Pointcnn: Convolution on x-transformed points. In: Advances in neural information processing systems, pp 820–830 1, 2, 3, 9, 11, 12, 13, 14

Liu Y, Fan B, Xiang S, Pan C (2019) Relation-shape convolutional neural network for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 8895–8904 9, 14

van der Maaten L, Hinton G (2008) Visualizing high-dimensional data using t-sne. Journal of Machine Learning Research 10

Manning CD, Raghavan P, Schütze H (2008) Introduction to Information Retrieval. Cambridge University Press, URL http://nlp.stanford.edu/IR-book/information-retrieval-book.html 10

Maturana D, Scherer S (2015) Voxnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 922–928 9

Mian A, Bennamoun M, Owens R (2010) On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. International Journal of Computer Vision 89(2-3):348–361 11

Poulenard A, Rakotosaona MJ, Ponty Y, Ovsjanikov M (2019) Effective rotation-invariant point cnn with spherical harmonics kernels. International Conference on 3D Vision 2, 3, 9

Qi CR, Su H, Nießner M, Dai A, Yan M, Guibas LJ (2016) Volumetric and multi-view cnns for object classification on 3d data. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5648–5656 2, 9

Qi CR, Su H, Mo K, Guibas LJ (2017a) Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 652–660 1, 2, 3, 6, 9, 12, 13, 14

Qi CR, Yi L, Su H, Guibas LJ (2017b) Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems, pp 5105–5114 1, 2, 3, 8, 9, 10, 12, 13, 14

Rao Y, Lu J, Zhou J (2019) Spherical fractal convolutional neural networks for point cloud recognition. In: Computer Vision and Pattern Recognition 2, 3, 8, 9, 13

Riegler G, Osman Ulusoy A, Geiger A (2017) Octnet: Learning deep 3d representations at high resolutions. In: Proceed-

ings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 3577–3586 2

Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention, pp 234–241 7

Savva M, Yu F, Su H, Aono M, Chen B, Cohen-Or D, Deng W, Su H, Bai S, Bai X, et al. (2016) Shrec16 track: largescale 3d shape retrieval from shapenet core55. In: Proceedings of the eurographics workshop on 3D object retrieval, vol 10 13

Su H, Maji S, Kalogerakis E, Learned-Miller E (2015) Multi-view convolutional neural networks for 3d shape recognition. In: Proceedings of the IEEE international conference on computer vision, pp 945–953 2, 9

Tatsuma A, Aono M (2009) Multi-fourier spectra descriptor and augmentation with spectral clustering for 3d shape retrieval. The Visual Computer 25(8):785–804 13

Thomas H (2020) Rotation-invariant point convolution with multiple equivariant alignments. In: 2020 International Conference on 3D Vision (3DV), pp 504–513 2, 3

Tombari F, Salti S, Di Stefano L (2010) Unique signatures of histograms for local surface description. In: Proceedings of the European Conference on Computer Vision, Springer, pp 356–369 4, 11

Uy MA, Pham QH, Hua BS, Nguyen DT, Yeung SK (2019) Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: International Conference on Computer Vision 1, 12, 13

Wang PS, Liu Y, Guo YX, Sun CY, Tong X (2017) O-cnn: Octree-based convolutional neural networks for 3d shape analysis. ACM Transactions on Graphics 36(4):1–11 2

Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM (2019) Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics 2, 9, 12, 13, 14

Weiler M, Hamprecht FA, Storath M (2018) Learning steerable filters for rotation equivariant cnns. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 849–858 3

Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J (2015a) 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1912–1920 1, 2, 3, 4

Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J (2015b) 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1912–1920 8, 13

Xu Y, Fan T, Xu M, Zeng L, Qiao Y (2018) Spidercnn: Deep learning on point sets with parameterized convolutional filters. In: Proceedings of the European Conference on Computer Vision, pp 87–102 1, 2, 14

Yi L, Kim VG, Ceylan D, Shen IC, Yan M, Su H, Lu C, Huang Q, Sheffer A, Guibas L (2016) A scalable active framework for region annotation in 3d shape collections. ACM Transactions on Graphics 35(6):1–12 1

Zaharescu A, Boyer E, Varanasi K, Horaud R (2009) Surface feature detection and description with applications to mesh matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp 373–380 11

Zhang Z, Hua BS, Rosen DW, Yeung SK (2019a) Rotation invariant convolutions for 3d point clouds deep learning. In: International Conference on 3D Vision, pp 204–213 2, 3, 4, 6, 9, 10, 11, 12, 13, 14

Zhang Z, Hua BS, Yeung SK (2019b) Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1607–1616 1, 2, 3, 6

Zhang Z, Hua BS, Chen W, Tian Y, Yeung SK (2020) Global context aware convolutions for 3d point cloud understanding. In: International Conference on 3D Vision 2, 3, 6, 9, 10, 13, 14

Zhao Y, Birdal T, Lenssen JE, Menegatti E, Guibas L, Tombari F (2020) Quaternion equivariant capsule networks for 3d point clouds. In: European Conference on Computer Vision, Springer, pp 1–19 1