The Visual Computer
https://doi.org/10.1007/s00371-023-02921-y

ORIGINAL ARTICLE

Unsupervised contrastive learning with simple transformation for 3D point cloud data

Jincen Jiang[1] · Xuequan Lu[2] · Wanli Ouyang[3] · Meili Wang[1]

Accepted: 28 May 2023

Abstract

Though a number of point cloud learning methods have been proposed to handle unordered points, most of them are supervised and require labels for training. By contrast, unsupervised learning of point cloud data has received much less attention to date. In this paper, we propose a simple yet effective approach for unsupervised point cloud learning. In particular, we identify a very useful transformation which generates a good contrastive version of an original point cloud. They make up a pair. After going through a shared encoder and a shared head network, the consistency between the output representations are maximized with introducing two variants of contrastive losses to respectively facilitate downstream classification and segmentation. To demonstrate the efficacy of our method, we conduct experiments on three downstream tasks which are 3D object classification (on ModelNet40 and ModelNet10), shape part segmentation (on ShapeNet Part dataset) as well as scene segmentation (on S3DIS). Comprehensive results show that our unsupervised contrastive representation learning enables impressive outcomes in object classification and semantic segmentation. It generally outperforms current unsupervised methods, and even achieves comparable performance to supervised methods.

Keywords Unsupervised contrastive learning · Point cloud · 3D object classification · Semantic segmentation

1 Introduction

Point cloud, as an effective representation for 3D geometric data, has attracted noticeable attention recently. It has been used for learning-based segmentation, classification, object detection, etc. Promising results have been achieved among those application fields. In this work, we focus on the use of 3D point clouds for classification and segmentation tasks. They respectively target to automatically recognize 3D objects and predict segment labels, which are crucial in multimedia computing, robotics, etc.

Most of existing methods for 3D point cloud analysis [1–9] use annotated data for training. Nevertheless, annotation is time-consuming and costly, especially for a considerable amount of data. In the real world, it is particularly challenging to have annotated data for training all the time. Unsupervised learning is a good alternative [10–13]. For example, Latent-GAN [10] used a deep architecture of Autoencoder (AE), and trained a minimal GAN in the AE's latent space for learning representations of point clouds. FoldingNet [11] proposed a new AE to get the codeword which can represent the high-dimensional embedding point cloud, and the fully-connected decoder was replaced with the folding-based decoder. MAP-VAE [12] conducted half-to-half predictions (splitting point cloud into a front half and a back half with several angles), and then combined them with global self-supervision to capture the geometry and structure of the point cloud. 3D-PointCapsNet [13] used the encoder-decoder structure, and concatenated the features from the encoder to form the point capsules. These methods usually employ the AE as the backbone, and often suffer from the curse of less quality representations. As a result, they may still induce less

✉ Meili Wang
wml@nwsuaf.edu.cn

Jincen Jiang
jinec@nwsuaf.edu.cn

Xuequan Lu
xuequan.lu@deakin.edu.au

Wanli Ouyang
wanli.ouyang@sydney.edu.au

1 College of Information Engineering, Northwest A&F University, Xianyang, China

2 School of Information Technology, Deakin University, Geelong, Australia

3 Shanghai Artifcial Intelligence Laboratory, Shanghai, China

desired performance on downstream tasks (e.g. classification and segmentation).

Motivated by the above analysis, we propose an unsupervised representation learning method which can facilitate the downstream 3D object classification and semantic segmentation. Our core idea is to maximize the agreement or consistency between the representations of the original point cloud and its transformed version (i.e. contrastive version).

We simply utilize one transformation to generate the transformed version point cloud and pair it with the original ones. And then feed them into a shared base encoder network (e.g. former part of PointNet [7] with global feature), followed by a subsequent projection head network (e.g. latter part of PointNet: several mlp layers). The agreement maximization is imposed on the outputs of the projection head network, to facilitate the training efficiency and better preserve the rich representations output from the encoder. Since there are no labels involved in training, it is unsupervised representation learning for 3D point cloud data.

To validate our unsupervised method, we conduct experiments for the object classification task on ModelNet40 and ModelNet10, the shape part segmentation task on ShapeNet Part dataset, and the scene segmentation task on the S3DIS dataset. Extensive results show that our unsupervised contrastive representative learning enables impressive outcomes in terms of the three tasks. Our method generally outperforms state-of-the-art unsupervised techniques, and is even comparable to certain supervised counterparts.

The contributions of this paper are:

- An unsupervised representation learning approach which is simple yet effective on 3D point cloud data,
- A simple transformation in generating a good contrastive version of an original point cloud, which is better than other complex transformations,
- Two variants of point cloud-based contrastive losses for downstream classification and segmentation, respectively,
- Experiments and analysis on three tasks (classification, shape part segmentation and scene segmentation), achieving promising performance with our simple contrastive learning.

2 Related work

Unlike 2D images, which consist of regular and uniform pixels, point cloud data are often irregular, sparse and contaminated with noise/outliers during the obtaining procedure of scanning and processing [14–17]. 3D point cloud learning techniques can be generally classified into three categories: (1) voxel based [1–3], (2) view based [4–6, 18, 19] and (3) point based [7–9, 20–27]. Voxel-based methods often involve resolution and memory issues, and view-based approaches are often criticized for the tedious pre-processing, i.e. projecting each 3D object onto 2D image planes. Point-based techniques are capable of learning features from point cloud data straightforwardly. In fact, most of these methods are supervised.

Voxel based techniques. 3D volumetric CNNs (Convolutional Neural Network) imitates classical 2D CNNs by performing voxelization on the input point cloud. 3D ShapeNets was designed for learning volumetric shapes [1]. Riegler et al. proposed OctNet for deep learning with sparse 3D data [2]. Wang et al. presented an Octree-based CNN for 3D shape analysis, which was called O-CNN [3]. These methods are proposed to improve 3D volumetric CNNs and reach high volume resolutions.

View based methods. View-based methods are to project 3D point cloud data onto the regular image planes. For example, MVCNNs used multiple images rendered from the 3D shapes to fit classical 2D CNNs [4]. Su et al. proposed to utilize a sparse set of samples in a high-dimensional lattice as the representation of a collection of points [18]. Zhou et al. proposed the multi-view saliency guided deep neural network (MVSG-DNN) which contains three modules to capture and extract the features of individual views to compile 3D object descriptors for 3D object retrieval and classification [19]. Xu et al. used a LSTM-based network to recurrently aggregate the 3D objects shape embedding from an image sequence and estimate images of unseen viewpoints, aiming at the fusion of multiple views' features [28]. Huang et al. devised a view mixture model (VMM) to decompose the multiple views into a few latent views for the descriptor construction [29]. Li et al. presented an end-to-end framework to learn local multi-view descriptors for 3D point clouds [5]. Lyu et al. projected 3D point clouds into 2D image space by learning the topology-preserving graph-to-grid mapping [6].

Point based methods. PointNet is a seminal work on point-based learning [7]. In PointNet, max-pooling operation is used to learn permutation-invariant features. The original authors introduced PointNet++, a hierarchical neural network that applied PointNet recursively on a nested partitioning of the input point set [8]. It achieved better learning outcomes than PointNet. Later, pointCNN was introduced to learn an X-transformation from the input points, to promote the weighting of the input features and the permutation of the points into a latent order [9]. PointConv, a density re-weighted convolution, was proposed to fully approximate the 3D continuous convolution on any set of 3D points [20]. Xu et al. proposed SpiderCNN to extract geometric features from point clouds [21]. Liu et al. designed a Relation-Shape Convolutional Neural Network to learn the geometric topology constraints among points [22]. Simonovsky et al. generalized the convolution operator from regular grids to arbitrary graphs and applied it to point cloud classification [30]. Parametric Continuous Convolution was introduced to exploit parameterized kernel functions that spanned the full continuous vector space [31]. Li et al. came up with a self-organizing network which applied hierarchical feature aggregation using self-organizing map [32]. It included a point cloud auto-encoder as pre-training to improve network performance. Komarichev et al. presented an annular convolution operator to better capture the local neighborhood geometry of each point by specifying the (regular and dilated) ring-shaped structures and directions in the computation [23]. Zhao et al. put forwarded PointWeb to enhance local neighborhood features for point cloud processing [33]. Xie et al. developed a new representation by adopting the concept of shape context as the building block and designed a model (ShapeContextNet) for point cloud recognition [34]. Wang et al. designed a new neural network module dubbed EdgeConv which acts on graphs dynamically computed in each layer [24]. More recently, Fujiwara et al. proposed to embed the distance field to neural networks [35]. Lin et al. defined learnable kernels with a graph max-pooling mechanism for their 3D Graph Convolution Networks (3D-GCN) [25]. Yan et al. presented the adaptive sampling and the local-nonlocal modules for robust point cloud processing [36]. Qiu et al. proposed a network considering both low-level geometric information of 3D space points explicitly and high-level local geometric context of feature space implicitly [37]. Chen et al. presented a hierarchical attentive pooling graph network (HAPGN) for segmentation which includes the gated graph attention network to get a better representation of local features and hierarchical graph pooling module to learn hierarchical features [38]. Liu et al. devised a point context encoding module (PointCE) and a semantic context encoding loss (SCE-loss) to capture the rich semantic context of a point cloud adaptively, achieving improved segmentation performance [39].

Unsupervised representation learning. Yang et al. proposed an autoencoder (AE), referred to as FoldingNet, for unsupervised learning on point cloud data [11]. MAP-VAE was proposed to enable the learning of global and local geometry by jointly leveraging global and local self-supervision [12]. Rao et al. presented bidirectional reasoning between the local structures and the global shape for unsupervised representation learning of point clouds [40]. It used a much larger RSCNN as backbone (4$\times$RSCNN) [22]. Zhang et al. presented an explainable machine learning method for point cloud classification by building local-to-global features through iterative one-hop information exchange, and feeding the feature vector to a random forest classifier for classification [41]. Different from them, we create a contrastive pair for each point cloud, and our framework simply consists of an encoder network and a head network. The encoder outputs global representations (features) for downstream networks and the head outputs projection features (a smaller size) for calculating the loss.

More recently, Xie et al. presented an unsupervised pre-training framework called PointContrast for high-level scene understanding tasks [42]. Their findings demonstrated that the learned representation could generalize across domains. [42] focused on 3D scenes (pretrained on a very large-scale generated dataset (about 1 terabyte), and sophisticatedly considered matched points (i.e. common points) of two different views (at least 30% overlap) as pairs. Unlike that, our point cloud level-based approach simply uses a rotational transformation to generate a transformed version of an original point cloud. It can easily get a great pose discrepancy, without requiring point cloud overlap to satisfy the demand of obtaining a certain number of matched points. In essence, a pair of matched points are treated as a pair in [42] to learn point-level features, while a pair of point clouds (a point cloud consisting of a series of points) are regarded as a pair in our work. Treating the point clouds as the pair in our method has the advantage of learning better global representations when compared with [42]. It is also intuitive and straightforward to use point cloud level, while PointContrast [42] can hardly obtain point cloud representations directly and is suitable for point-wise tasks, e.g., scene segmentation. In comparison, our global feature of point cloud level can be easily used in both point cloud level and point-wise tasks (e.g., classification and segmentation). Meanwhile, for unsupervised learning we derive two variants of contrastive losses based on point clouds, which respectively facilitate two different types of downstream tasks (i.e., classification and segmentation). Finally, the backbone networks are different: they use a Sparse Residual U-Net which requires voxelization of point clouds, while we use a simple encoder-head structure.

Difference from SimCLR [43]: It is derived on top of SimCLR [43], but is largely different from SimCLR: (1) SimCLR is designed for 2D images, and our contrastive learning is for 3D point cloud data, which involves irregular point distribution, and poses more challenges than 2D images with regular grids. (2) SimCLR uses multiple transformations for an original image, and these two different versions of images act as a pair. By contrast, we only generate a transformed version of the original point cloud with a simple transformation (e.g. rotation), thus forming a contrastive pair of this point cloud (i.e. a pair in point cloud level). (3) SimCLR typically uses TPU resources with a large batch size for training, while we demonstrate an elegant approach for 3D point cloud representation learning, which is simple yet effective, without requiring TPU computing resources.

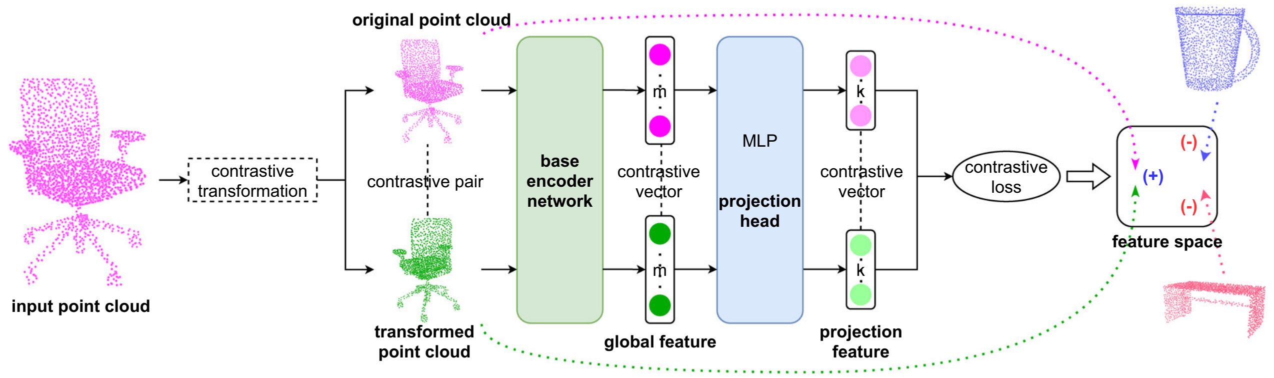

Fig. 1 Overview of our unsupervised contrastive representation learning method. Given a point cloud, the transformation (rotation in this work) is used to get a transformed version of the original point cloud, which defines a pair. Then, the pairs are input to the base encoder network (e.g. PointNet or DGCNN) to learn the global feature of each model. The projection head is further used to reduce the global feature dimension and for effective loss degradation. The contrastive loss encourages a pair of point clouds to be consistent in the feature space

3 Method

In this work, we take 3D object classification and semantic segmentation (shape and scene) as the downstream tasks of our unsupervised contrastive representation learning. To clearly elaborate our method, we take downstream object classification as an example when designing the unsupervised stage, and we will later explain how to extend it to shape and scene segmentation.

Given an unlabeled point cloud, we first use a transformation (i.e. rotation) to generate its transformed version, thus constructing a contrastive point cloud pair for this original point cloud. They are fed into a base encoder network, in order to learn a pair of global features or representations. The global features are then passed to a projection head network, to obtain another pair of representations. These two representations from the projection head network target to reach a maximum agreement with the aid of a contrastive loss function. Figure 1 shows the framework of our unsupervised contrastive representation learning.

3.1 Unsupervised contrastive representation learning

Contrastive transformation. Unlike 2D images, point cloud data often have an irregular distribution in 3D space, and have a complex degree of freedom. Given this, it is more difficult to identify the practically useful transformations for constructing a good contrastive pair of a point cloud. SimCLR [43] utilized different types of transformations for a single image (e.g. cropping, rotation), and generated two transformed versions. In contrast, we only use one transformation for simplicity, and pair the original point cloud with the transformed version, that is, a pair including the original point cloud and its transformed counterpart. Common transformations in 3D space are rotation, cutout, crop, scaling, smoothing, noise corruption, etc. Since heavy noise corruption will destroy the object shapes, we exclude this for transformations applied here. Jittering is analogous to light noise, and we use jittering for data augmentation, following the protocol of the state-of-the-art point-based methods. In this work, we select rotation as the transformation, and use it to generate a transformed version for the original point cloud. We provide the discussion of the choice in ablation studies (Sect. 4.6).

Base encoder network. Point-based networks, such as PointNet [7], DGCNN [24], Pointfilter [14], often involve a pooling layer to output the global feature for an input point cloud. The former part of a point-based network before this layer (inclusive) can be naturally viewed as a based encoder in our framework. In other words, the input point cloud can be encoded into a latent representation vector (i.e. the global feature). In this sense, we can simply extract this former part of any such point-based networks as a base encoder network in our unsupervised contrastive representation framework. In this work, we select some state-of-the-art point-based networks including PointNet and DGCNN as the backbone, and extract their former parts as our base encoder accordingly. It is interesting to discover that the encoders involving T-Net (i.e. transformation net) will hinder the learning of unsupervised contrastive representations. We deduce that T-Net accounts for various rotated point cloud augmentations, which degrades the ability of capturing a large contrast between the input pair. As such, we remove the original T-Net (i.e. transformation net) in these encoders, if involved. We show the results of different encoders in Sect. 4.

Projection head network. Point-based networks usually have several fully connected layers to bridge the global feature with the final k-class vector. Similar to the encoder, we can also simply extract the latter part of a point-based network as the projection head. Alternatively, it is also flexible to customize a projection head network by designing more or fewer fully connected layers.

Mathematically, the final k-class vector (or representation vector) can be formulated as

$$\mathbf{z_i} = H(E(\mathbf{P})),$$
$$\mathbf{z_j} = H(E(\mathbf{P}')), \quad (1)$$

where $\mathbf{P}$ is an original point cloud and $\mathbf{P}'$ is its transformed counterpart. E and H denote the encoder network and the projection head network, respectively.

Contrastive loss function. We first randomly select n samples, and use the selected transformation (i.e. rotation) to generate another n corresponding transformed counterparts, resulting in n pairs ($2n$ samples) constituting the minibatch. Analogous to SimCLR [43], we also do not explicitly define positive or negative pairs. Instead, we select a pair as the positive pair, and the remaining $(n-1)$ pairs (i.e. $2(n-1)$ samples) are simply regarded as negative pairs.

As for the unsupervised loss function, InfoNCE [44] is a widely-used loss function for unsupervised representation learning of 2D images. More recently, [42] also utilized a similar loss for contrastive scene representation learning. Inspired by them, we also introduce a variant as our unsupervised loss function, which is defined as

$$L = -\frac{1}{S} \sum_{(i,j)\in S} \log \frac{\exp(\mathbf{z_i} \cdot \mathbf{z_j}/\tau)}{\sum_{(\cdot,t)\in S, t\neq j} \exp(\mathbf{z_i} \cdot \mathbf{z_t}/\tau)}, \quad (2)$$

where S is the set of all positive pairs (point cloud level), and τ is a temperature parameter. $||$ denotes the cardinality of the set. The loss is computed using all the contrastive pairs and is equivalent to applying the cross entropy with pseudo labels (e.g. $0 \sim 15$ for 16 pairs). We found it works very well in our unsupervised contrastive representation learning.

3.2 Downstream 3D object classification

We take 3D object classification as the first downstream task in this work, to validate our unsupervised representation learning. The above designed scheme is immediately ready for the unsupervised representation learning to facilitate the downstream classification task. In particular, we will utilize two common schemes for validation here. One is to train a linear classification network by taking the learned representations of our unsupervised learning as input. Here, the learned representation is the global feature. We did not choose the k-class representation vector as it had less discriminative features than the global feature in our framework, and it induced a poor performance (see Sect. 4.6). The other validation scheme is to initialize the backbone with the unsupervised trained model and perform a supervised training. We will demonstrate the classification results for these two validation schemes in Sect. 4.

3.3 Downstream semantic segmentation

To further demonstrate the effectiveness of our unsupervised representation learning, we also fit the above unsupervised learning scheme to the downstream semantic segmentation, including shape part segmentation and scene segmentation. Since it is a different task from 3D object classification, we need to design a new scheme to facilitate unsupervised training. We still use the rotation to generate a transformed version of an original point cloud (e.g. a shape point cloud or a split block from the scene), and view them as a contrastive pair (i.e. point cloud level). As for segmentation, each point in the point cloud has a feature representation. For unsupervised representation learning, we compute the mean of all point-wise cross entropy to evaluate the overall similarity within the mini-batch. We therefore define a loss function for semantic segmentation as:

$$L = -\frac{1}{S} \sum_{(a,b)\in S} \frac{1}{P_{(a,b)}} \sum_{(i,j)\in P_{(a,b)}} \log \frac{\exp(\mathbf{z_i} \cdot \mathbf{z_j}/\tau)}{\sum_{(\cdot,t)\in P_{(a,b)}, t\neq j} \exp(\mathbf{z_i} \cdot \mathbf{z_t}/\tau)}, \quad (3)$$

where S is the set of all positive pairs (i.e. point cloud a and b), and $P_{(a,b)}$ is the set of all point pairs (i.e. the same point id) of the point cloud a and b. Similarly, we apply the cross entropy with pseudo labels which match the point indices (e.g. $0 \sim 2047$ for 2048 points).

4 Experimental results

4.1 Datasets

Object classification. We utilize ModelNet40 and ModelNet10 [1] for 3D object classification. We follow the same data split protocols of PointNet-based methods [7, 8, 24] for these two datasets. For ModelNet40, the train set has $9,840$ models and the test set has $2,468$ models, and the dataset consists of 40 categories. For ModelNet10, $3,991$ models are for training and 908 models for testing. It contains 10 categories. For each model, we use $1,024$ points with only (x, y, z) coordinates as the input, which is also consistent with previous works.

Note that some methods [10, 11, 13, 45] are pre-trained under the ShapeNet55 dataset [47]. We also conduct a version

Table 1 Classification results of unsupervised methods and our method (Linear Classifier), on the datasets of ModelNet40 and ModelNet10

Methods	Pretrained dataset	Input data	Resolution e.g. # points	ModelNet40 accuracy	ModelNet10 accuracy
Latent-GAN [10]	ShapeNet55	*xyz*	2k	85.70	95.30
FoldingNet [11]	ShapeNet55	*xyz*	2k	88.40	94.40
MRTNet [45]	ShapeNet55	*xyz*	Multi-resolution	86.40	–
3D-PointCapsNet [13]	ShapeNet55	*xyz*	2k	88.90	–
Ours (DGCNN)	ShapeNet55	*xyz*	2k	89.37	–
VIPGAN [46]	ModelNet40	Views	12	91.98	94.05
Latent-GAN [10]	ModelNet40	*xyz*	2k	87.27	92.18
FoldingNet [11]	ModelNet40	*xyz*	2k	84.36	91.85
3D-PointCapsNet [13]	ModelNet40	*xyz*	1k	87.46	–
PointHop [41]	ModelNet40	*xyz*	1k	89.10	–
MAP-VAE [12]	ModelNet40	*xyz*	2k	90.15	94.82
GLR (RSCNN-Large) [40]	ModelNet40	*xyz*	1k	92.9	–
Ours (PointNet [7])	ModelNet40	*xyz*	1k	88.65	90.64
Ours (DGCNN [24])	ModelNet40	*xyz*	1k	90.32	95.09

Both ShapeNet55 (upper part) and ModelNet40 (bottom part) pretrained datasets are provided

of ShapeNet55 training for the classification task. We used the same dataset as [13], which has $57,448$ models with 55 categories, and all models will be used for unsupervised training. Following the same setting of previous work, we use 2048 points as input.

We provide comparison experiments with PointContrast [42] for the classification task, and they use the ShapeNet-Core [47] for finetuning. The dataset contains 51,127 pre-aligned shapes from 55 categories, which has 35,708 models for training, 5158 models for validation and 10,261 models for testing. We use 1024 points as input which is the same as PointContrast [42].

Shape part segmentation. We use the ShapeNet Part dataset [48] for shape part segmentation, which consists of 16,881 shapes from 16 categories. Each object involves 2 to 6 parts, with a total number of 50 distinct part labels. We follow the official dataset split and the same point cloud sampling protocol as [47]. Only the point coordinates are used as input. Following [8, 24], we use mean Intersection-over-Union (mIoU) as the evaluation metric.

Scene segmentation. We also evaluate our model for scene segmentation on Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS) [49]. This dataset contains 3D scans of 271 rooms and 6 indoor areas, covering over 6000m^2. We follow the same setting as [8, 24]. Each room is split with $1\text{m} \times 1$ m area into little blocks, and we sampled 4096 points of each block. Each point is represented as a 9D vector, which means the point coordinates, RGB color and normalized location for the room. Each point is annotated with one of the 13 semantic categories. We also follow the same protocol of adopting the sixfold cross validation for the six area.

Please refer to the Appendices for additional information and visual results.

4.2 Experimental setting

We use Adam optimizer for our unsupervised representation training. We implemented our work with TensorFlow, and use a single TITAN V GPU for training (DGCNN using multiple GPUs).

For downstream 3D object classification on ModelNet40, ModelNet10, ShapeNet55 and ShapeNetCore, we use a batch size of 32 (i.e. 16 contrastive pairs) for training and testing. Temperature hyper-parameter τ is set as 1.0. We use the same dropouts with the original methods accordingly, i.e. 0.7 for PointNet as backbone, 0.5 for DGCNN as backbone. The initial decay rate of batch normalization is 0.5, and will be increased no larger than 0.99. The training starts with a 0.001 learning rate, and is decreased to 0.00001 with an exponential decay.

We employ DGCNN as the backbone for semantic segmentation. As for shape part segmentation on ShapeNet Part dataset, we utilize a batch size of 16 (i.e. 8 constrasive pairs) for training. We use a batch size of 12 (i.e. 6 constrasive pairs) for scene segmentation on S3DIS. For the two tasks, we simply use a batch size of 1 during testing, and the other settings follow DGCNN.

4.3 3D object classification

We conduct two kinds of experiments to evaluate the learned representations of our unsupervised contrastive learning. We first train a simple linear classification network with the

Table 2 Comparison results of classification accuracy with limited training data (different ratios)

Methods	10%	20%	30%
FoldingNet [11])	81.2	83.6	84.2
Ours (DGCNN as backbone)	80.6	85.4	86.6

unsupervised representations as input. Secondly, we take our unsupervised representation learning as pretraining, and initialize the weights of the backbone before supervised training. Tables 1 and 3 show 3D object classification results for our method and a wide range of state-of-the-art techniques.

Linear classification evaluation. In this part, we use the former part of PointNet [7] and DGCNN [24] as the base encoder, and use the latter mlp layers as the projection head. The learned features are used as the input for training the linear classification network. We use the test accuracy as the evaluation of our unsupervised contrastive learning. Comparisons are reported in Table 1. Regarding linear classification evaluation, our method with DGCNN as backbone always performs better than our method using PointNet as backbone, for example, 95.09% versus 90.64% for ModelNet10, 90.32% versus 88.65% for ModelNet40. This is due to a more complex point-based structure of DGCNN. Our method with DGCNN as backbone also outperforms most unsupervised techniques, like two recent methods PointHop (1.22% gain) and MAP-VAE (0.17% gain), and is comparable to some supervised methods in Table 3, for example, 90.32% versus 90.6% (O-CNN) and 90.9% (SO-Net with *xyz*) on ModelNet40, 95.09% versus 93.9% (supervised 3D-GCN) on ModelNet10. The GLR [40] mined rich semantic and structural information, and used a larger RSCNN as backbone (4×RSCNN) [22], resulting in a better accuracy than our method.

Notice that some methods used a larger ShapeNet55 dataset for training [10, 11, 13, 45]. Although the previous work [12] re-implemented them by training on ModelNet40, they use 2048 points rather than our 1024. To provide additional insights, we re-implement and train a state-of-the-art method (3D-PointCapsNet [13]) on ModelNet40 with 1024 points, and train a linear classifier for evaluation. We choose this method since its code is publicly available and it is recent work. From Table 1, it is obvious that our method (DGCNN as backbone) still outperforms 3D-PointCapsNet by a 2.86% margin.

To show our learned representations have the transfer capability, we also train our method (DGCNN as backbone) on ShapeNet55 dataset for unsupervised contrastive learning and then feed the ModelNet40 dataset to the trained model to get point cloud features. We use these features to train a linear classifier on ModelNet40 for evaluation. From Table 1 we can see that our method achieves the best result compared with other state-of-art methods on the same settings (Latent-GAN [10], FoldingNet [11], and 3D-PointCapsNet [13]), exceeding them by 3.67%, 0.97%, and 0.47%, respectively.

We also conduct an experiment for training with limited data to verify the capability of our pretrained model with linear classifier. The results can be seen in Table 2. Our pretrained model achieves 86.6% with 30% data, which is 2.4% higher compared with FoldingNet's [11] 84.2%. With less data, e.g. 10% of the data, our result is 0.6% lower than FoldingNet. We suspect that with far fewer data, our contrastive learning method could less effectively learn the features of each sample with such simple transformation, and result in less improvement.

Pretraining evaluation. In addition to the above evaluation using a linear classifier, we further utilize the pre-training evaluation to demonstrate the efficacy of our unsupervised contrastive representation learning. Specifically, we also select PointNet and DGCNN as the backbone, in which the part before and including the global feature is regarded as the base encoder, and the remaining classification branch (i.e. several mlp layers) as the projection head. After our unsupervised representation training, we initialize the corresponding network with the unsupervised trained model, and then perform the supervised training. Table 3 shows the comparison results of our method and the state-of-the-art 3D object classification techniques with supervised training.

The pretraining evaluation based on our unsupervised representation learning sees an improvement over the original backbone network, increased from 89.2% to 90.44% (1.24% increase) with PointNet and from 92.2% to 93.03% (0.83% increase) with DGCNN on ModelNet40. Regarding ModelNet10, the accuracy of PointNet as our backbone for pretraining evaluation is 94.38%, which is on par with the supervised 3D-GCN (93.9%). It is interesting to see that our method (DGCNN as backbone) is the best one on ModelNet10 in the pretraining evaluation, while the second best is achieved by two very recent supervised methods (Neural Implicit [35] and PointASNL [36]). [35] even used a large weight matrix as the input for classification training. For ModelNet40, our method (DGCNN as backbone) achieves 93.03%, outperforming almost all techniques including both supervised and unsupervised ones. For example, our method in this case outperforms the very recent supervised methods including 3D-GCN [25], Neural Implicit [35] and PointASNL [36]. Compared to using PointNet as backbone, taking DGCNN as backbone achieves a better classification accuracy, for example, 95.93% versus 94.38%, 93.03% versus 90.44%. Similarly, we believe this is mainly because DGCNN exploits richer information than PointNet.

Table 3 Classification results of other supervised methods and our method (Pretraining), on the datasets of ModelNet40 and ModelNet10

Methods	Input data	Resolution e.g. # points	ModelNet40 accuracy	ModelNet10 accuracy
Kd-Net (depth=10) [50]	Tree	$2^{10} \times 3$	90.6	93.3
PointNet++ [8]	*xyz*	1k	90.7	–
KCNet [51]	*xyz*	1k	91.0	94.4
MRTNet [45]	*xyz*	1k	91.2	–
SO-Net [32]	*xyz*	2k	90.9	94.1
KPConv [52]	*xyz*	6.8k	92.9	–
PointNet++ [8]	*xyz*, normal	5k	91.9	–
SO-Net [32]	*xyz*, normal	5k	93.4	–
O-CNN [3]	*xyz*, normal	–	90.6	–
PointCNN [9]	*xyz*	1k	92.2	–
PCNN [53]	*xyz*	1k	92.3	94.9
Point2Sequence [54]	*xyz*	1k	92.6	95.3
RS-CNN (voting) [22]	*xyz*	1k	93.6	–
Neural Implicit [35]	Weights	1024×256	92.2	95.7
PointASNL [36]	*xyz*	1k	92.9	95.7
3D-GCN [25]	*xyz*	1k	92.1	93.9
HAPGN [38]	*xyz*	1k	91.7	–
MVSG-DNN [19]	views	12	92.3	94.0
Scratch (PointNet [7])	*xyz*	1k	89.2	–
Ours (PointNet [7])	*xyz*	1k	90.44 (+1.24)	94.38
Scratch (DGCNN [24])	*xyz*	1k	92.2	–
Ours (DGCNN [24])	*xyz*	1k	93.03 (+0.83)	95.93

We distinguish the results of other methods from our method by the line

PointContrast [42] also presented an unsupervised contrastive learning approach, which is based on point level while ours is based on point cloud level. They validated their effectiveness on some datasets using the pretrain-finetuning strategy. In order to provide a potential comparison with it, we also used the ShapeNetCore dataset for the classification task with pretraining evaluation. The comparison results are shown in Table 4, and we can see that our method (DGCNN as backbone) outperforms them by 0.5%, though PointContrast is pretrained on a rather larger dataset (ScanNet). Note that our method is not suitable to be pretrained on ScanNet since this downstream task is for classification (requiring point cloud level features for classification) while ScanNet has point-wise labels. The good performance of our method is mainly due to the proper design of point cloud level-based contrastive pairs and contrastive learning, so that we can directly obtain the global feature from contrastive representation learning. We also re-implement DGCNN [24] on the ShapeNetCore dataset, which further demonstrates the effectiveness of our method by increasing from 84.0% (original DGCNN) to 86.2%. In comparison with PointContrast which improved 0.6% from the version of training from scratch, we achieve 2.2% increase.

4.4 Shape part segmentation

In addition to the 3D object classification, we also verify our method on shape part segmentation. The segmentation results are listed in Table 5. Here we take DGCNN as the backbone of our approach and simply employ the linear classifier evaluation setting. It can be seen from the table that our method in linear classification evaluation achieves 79.2% instance mIOU and 75.5% class mIOU, which are remarkably better than state-of-the-art unsupervised techniques including MAP-VAE [12] and Multi-task [55]. Specifically, our method outperforms MAP-VAE [12] and Multi-task [55] by a margin of 7.55% and 3.4%, respectively, in terms of class mIOU. Figure 2 illustrates some examples of our method (Linear Classifier setting) on the task of shape part segmentation.

4.5 Scene segmentation

We also test our method for the scene segmentation task on the S3DIS dataset, which typically appears to be more challenging than the shape part segmentation. Similarly, we utilize DGCNN as the backbone and adopt the Linear Classifier evaluation setting. We are not able to compare our method with unsupervised methods like MAP-VAE [12]

Table 4 Comparison results of PointContrast and our method on the dataset of ShapeNetCore with Pretraining evaluation

Methods	Accuracy
Trained from scratch (PointContrast [42])	85.1
PointContrast* [42]	85.7
Trained from scratch (original DGCNN [24])	84.0
Ours (DGCNN as backbone)	86.2

Note that * represents that the model is trained on ScanNet

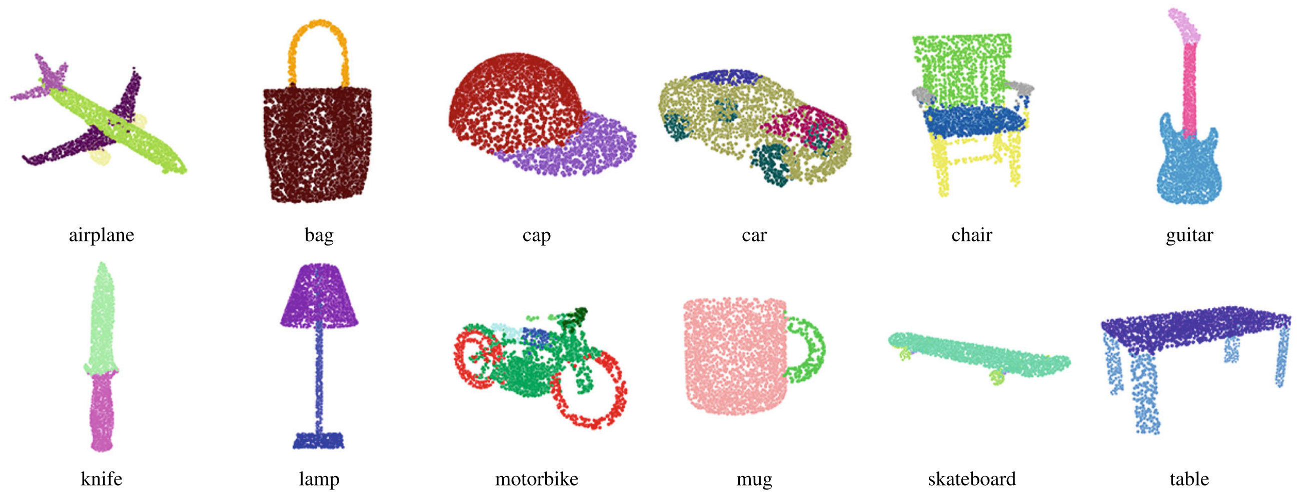

Fig. 2 Some examples of shape part segmentation using our method (Linear Classifier setting)

and Multi-task [55], since they did not provide scene segmentation results and their source codes are not publicly available. Table 6 lists the comparisons of onefold testing on Area 5. It is observed that our method even outperforms the supervised PointNet in terms of mean accuracy. Due to the unsupervised property, our method is inferior to the supervised PointCNN and fine-tuned PointContrast. Our method has relatively smaller mean IOU, which is probably due to the imbalanced categories and the limited minibatch size. The performance could be further improved if more powerful computing resources are allowed. Figure 3 shows a visual example for scene segmentation.

4.6 Ablation studies

Transformation. One of the key elements in our unsupervised representation learning is using 180° rotation around the Y axis to get the transformation. To comprehensively study the influence of transformation on representations, we consider many common transformations including rotation, cutout, crop, scale, jittering and smoothing. Figure 4 visualizes different transformations for a point cloud.

We list the comparison results of the above transformations in Table 7. It can be clearly observed that our choice attains the best accuracy, which is unlike SimCLR [43] that utilizes two different transformations of an image as the pair. We suspect that rotation is a very simple and effective transformation for 3D point cloud data, and a larger valid rotation would generate a greater pose discrepancy (i.e., contrast) in 3D space. As such, our choice using 180° rotation around the Y axis is better than others.

Furthermore, we apply two sequential transformations on one point cloud and make the result as a pair with the original point cloud. We chose the best transformation (i.e. rotate 180° around Y axis) as the first transformation, and then apply one of the rest of the transformations as the second. For more complex transformations, we group all the transformations into four categories, including rotation, scaling, jittering/smoothing, and cropping/cutout, and apply these four categories on one point cloud sequentially. We show the results in Table 8. It can be seen that after applying the complex transformation, it is still not as good as the best choice shown above. We suspect that the complex transformation may damage the information on the point cloud, thus leading to inferior results. Again, this verifies that our choice is the best transformation for 3D point cloud data in generating contrastive pairs.

Output of encoder versus output of projection head. We also compare the choices of using the output of the base encoder (i.e. global feature) and the output of the projection head for subsequent linear classification. Table 9 shows the comparison results of the two choices on ModelNet40 and

Table 5 Shape part segmentation results of our method (Linear Classifier) and state-of-the-art techniques on ShapeNet Part dataset

Methods	Supervised	Class mIOU	Instance mIOU	Air	Bag	Cap	Car	Chair	Ear	Guit	Kni	Lam	Lap	Mot	Mug	Pist	Rock	Ska	Tab
Kd-Net [50]	Yes	77.4	82.3	80.1	74.6	74.3	70.3	88.6	73.5	90.2	87.2	81.0	84.9	87.4	86.7	78.1	51.8	69.9	80.3
MRTNet [45]	yes	79.3	83.0	81.0	76.7	87.0	73.8	89.1	67.6	90.6	85.4	80.6	95.1	64.4	91.8	79.7	87.0	69.1	80.6
PointNet [7]	Yes	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
KCNet [51]	Yes	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
RS-Net [56]	Yes	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
SO-Net [32]	Yes	81.0	84.9	82.8	77.8	88.0	77.3	90.6	73.5	90.7	83.9	82.8	94.8	69.1	94.2	80.9	53.1	72.9	83.0
PointNet++ [8]	Yes	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
DGCNN [24]	Yes	82.3	85.2	84.0	83.4	86.7	77.8	90.6	74.7	91.2	87.5	82.8	95.7	66.3	94.9	81.1	63.5	74.5	82.6
KPConv [52]	Yes	85.1	86.4	84.6	86.3	87.2	81.1	91.1	77.8	92.6	88.4	82.7	96.2	78.1	95.8	85.4	69.0	82.0	83.6
Neural Implicit [35]	Yes	–	85.2	84.0	80.4	88.0	80.2	90.7	77.5	91.2	86.4	82.6	95.5	70.0	93.9	84.1	55.6	75.6	82.1
3D-GCN [25]	Yes	82.1	85.1	83.1	84.0	86.6	77.5	90.3	74.1	90.9	86.4	83.8	95.6	66.8	94.8	81.3	59.6	75.7	82.8
HAPGN [38]	Yes	87.1	89.3	87.1	85.7	90.1	86.2	91.7	78.3	94.3	85.9	82.6	95.2	77.9	94.3	90.1	73.9	90.3	90.6
MAP-VAE [12]	No	67.95	–	62.7	67.1	73.0	58.5	77.1	67.3	84.8	77.1	60.9	90.8	35.8	87.7	64.2	45.0	60.4	74.8
Multi-task [55]	No	72.1	77.7	78.4	67.7	78.2	66.2	85.5	52.6	87.7	81.6	76.3	93.7	56.1	80.1	70.9	44.7	60.7	73.0
Ours (DGCNN)	No	75.5	79.2	76.3	76.6	82.5	65.8	85.9	67.1	86.6	81.3	79.2	93.8	55.8	92.8	73.5	53.1	61.3	76.6

We distinguish between supervised and unsupervised learning methods by the line

Table 6 Scene segmentation results of our method (Linear Classifier) and some state-of-the-art techniques on testing Area 5 (Fold 1) of the S3DIS dataset

Methods	Supervised	Mean accuracy	Mean IOU
PointNet [7]	Yes	49.0	41.1
PointCE [39]	Yes	–	51.7
PointCNN [9]	Yes	63.9	57.3
PointContrast [42]	Yes	76.9	70.3
Ours (DGCNN)	No	59.4	32.6

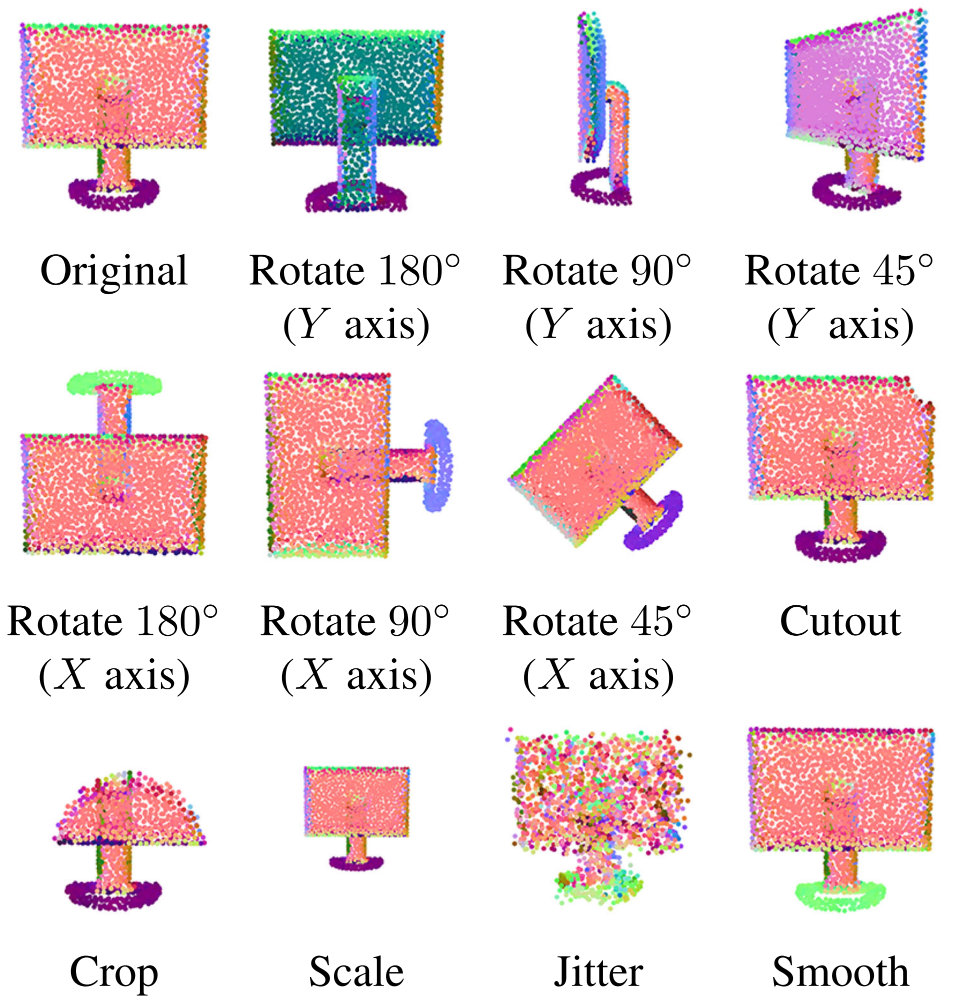

Fig. 4 Illustration for transformations used in Table 7

ModelNet10. We see that the former choice is better than the latter choice. We think the output of the base encoder involves more discriminative features for the training of the linear classifier.

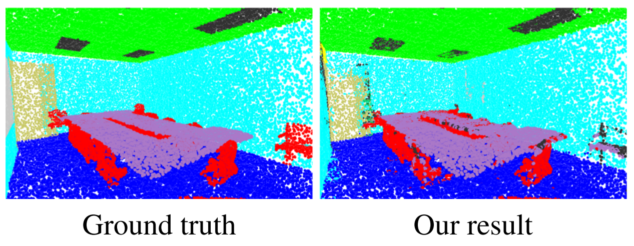

Fig. 3 Visual result of scene segmentation

Cross validation. In addition to the above evaluations, we further test the abilities of our unsupervised contrastive representation learning in a crossed evaluation setting. To achieve this, we use the learned representations from the unsupervised trained model on ModelNet40 to further train a linear classifier on ModelNet10, and vice versa. Classification outcomes are reported in Table 10. It can be observed that our unsupervised representation learning is indeed working in the cross-dataset setting. It also reveals that our unsupervised method trained on a large dataset would probably benefit the

Table 7 Comparison of different contrastive transformation on ModelNet10. DGCNN [24] is the backbone. We use linear classification evaluation for comparisons. % is used for classification accuracy

Transformation	Mean class accuracy	Overall accuracy
Rotate 180°(Y axis)	94.88	95.09
Rotate 90°(Y axis)	94.12	94.53
Rotate 45°(Y axis)	94.09	94.20
Rotate 180°(X axis)	93.21	93.53
Rotate 90°(X axis)	93.30	93.42
Rotate 45°(X axis)	93.71	93.97
Cutout	94.01	93.97
Crop	93.80	94.31
Scale	94.10	94.20
Jitter	93.95	93.97
Smooth	93.93	94.08

Table 8 Comparison of more complex contrastive transformation on ModelNet10. We distinguish between the results of using only two transformations and those of using four transformations (more complex) by the line. "Rotate" means 180° rotation around the Y axis. DGCNN [24] is the backbone. We use linear classification evaluation for comparisons. % is used for classification accuracy

Transformation	Mean class accuracy	Overall accuracy
Rotate + cutout	93.43	93.64
Rotate + crop	93.90	93.97
Rotate + scale	94.11	94.08
Rotate + jitter	94.33	94.42
Rotate + smooth	93.41	93.64
Rotate + scale + jitter + cutout	94.07	94.42
Rotate + scale + jitter + crop	93.92	94.31
Rotate + scale + smooth + cutout	93.56	93.86
Rotate + scale + smooth + crop	93.78	94.20
Rotate 180°(Y axis)	94.88	95.09

Table 9 Comparison of using the output of encoder and projection head for linear classification evaluation. PointNet [7] is the backbone. % is used for classification accuracy

Component	Dataset	Mean class accuracy	Overall accuracy
encoder	ModelNet40	83.81	88.65
head	ModelNet40	68.55	75.81
encoder	ModelNet10	90.55	90.64
head	ModelNet10	81.57	82.59

Table 10 Cross validation for ModelNet40 and ModelNet10. We perform unsupervised learning on one dataset and conduct the classification task on another dataset. PointNet [7] is the backbone. % is used for classification accuracy

Unsupervised dataset	Classification dataset	Mean class accuracy	Overall accuracy
ModelNet40	ModelNet10	90.00	90.51
ModelNet10	ModelNet40	77.13	82.87

Table 11 Pretraining validation to determine whether using projection head for initialization. ModelNet40 and ModelNet10 are used for datasets. PointNet [7] and DGCNN[24] are the backbone. % is used for classification accuracy

Backbone	Dataset	Head initialization	Mean class accuracy	Overall accuracy
PointNet	ModelNet10	Yes	93.80	93.97
PointNet	ModelNet10	No	94.23	94.38
PointNet	ModelNet40	Yes	86.64	90.22
PointNet	ModelNet40	No	86.80	90.44
DGCNN	ModelNet10	Yes	95.05	95.09
DGCNN	ModelNet10	No	95.78	95.93
DGCNN	ModelNet40	Yes	88.58	91.96
DGCNN	ModelNet40	No	89.52	93.03

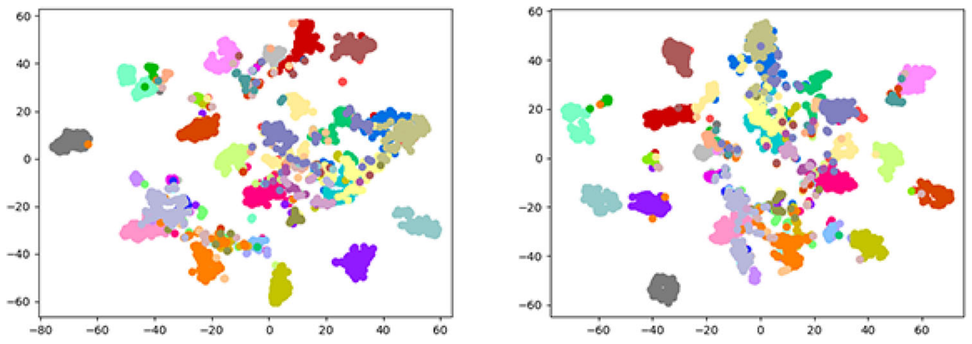

Fig. 5 T-SNE visualization of features. **a** without contrastive learning, **b** with contrastive learning

testing on another dataset greatly. In here, our method trained on ModelNet40 enables a better cross-test accuracy, compared to unsupervised training on ModelNet10 and testing on ModelNet40.

Pretraining evaluation: initializing projection head. Projection head is very useful in maximizing the agreement between the contrastive pair. However, it may hinder pretraining evaluation, if the corresponding part is initialized with the projection head of the unsupervised model. Table 11 shows that initializing encoder only produces better classification accuracy for PointNet/DGCNN on ModelNet10/ModelNet40, which confirms the judgement that initializing encoder only is a better choice.

T-SNE Visualization. We utilize the t-SNE to visualize the features learned on ModelNet40 in Fig. 5. It can be seen that using only a linear classifier can separate different features to some extent. It is worth noting that our model using unsupervised contrastive learning can better separate features after training a linear classifier, which implies that our contrastive learning is useful and effectively facilitates the classification task.

5 Conclusion

We have presented an unsupervised representation learning method for 3D point cloud data. We identified that rotation is a very useful transformation for generating a contrastive version of an original point cloud. Unsupervised representations are learned via maximizing the correspondence between paired point clouds (i.e. an original point cloud and its contrastive version). Our method is simple to implement and does not require expensive computing resources like TPU. We evaluate our unsupervised representations for the downstream tasks including 3D object classification, shape part segmentation and scene segmentation. Experimental results demonstrate that our method generates impressive performance. In the future, We would like to exploit semi-supervised techniques like [57] to improve the performance. We would also like to extend our approach to other interesting applications such as 3D object detection.

Declarations

Conflict of interest The authors declare that the work is original and has not been submitted elsewhere.

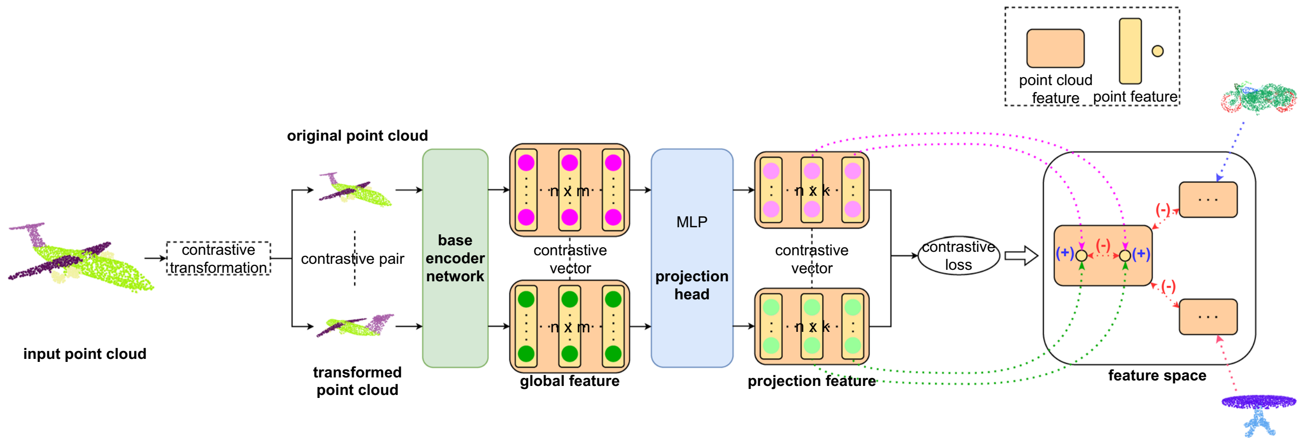

Fig. 6 Overview of our unsupervised contrastive learning for the downstream segmentation task. All the point clouds in the minibatch will be mapped into the feature space. The designed contrastive loss (shown in the main paper) encourages a pair of point clouds (original point cloud and its transformed point cloud) to be consistent in the feature space, and the point-wise features of the same point ID also tend to be consistent

Appendix A Overview of segmentation

We also achieve the task of point cloud semantic segmentation, including shape part segmentation and scene segmentation. Different from the 3D object classification task, we need to gain all the point-wise features in the point cloud, which is the key to solve the segmentation task. For our unsupervised contrastive learning, as shown in Fig. 6, we still consider the original point cloud and its transformed point cloud as a contrastive pair. However, in order to ensure that the feature of each point in the point cloud will be learned, we use the mean of point-wise cross entropy to evaluate the point cloud similarity, and try to maximize the similarity of the positive pair (all other pairs of point clouds in the minibatch are viewed as negative pairs). In this unsupervised manner, our framework can learn the feature of each point in the point cloud.

Appendix B Additional visual results on scene segmentation

In this section, we show more visual results on scene segmentation. Similarly, we utilize the Linear Classifier setting for this downstream task. Figure 7 shows the visual results of several scenes. We can observe from the figure that our method produces close segmentation results to the ground truth. This demonstrates the capability of our unsupervised representation learning method.

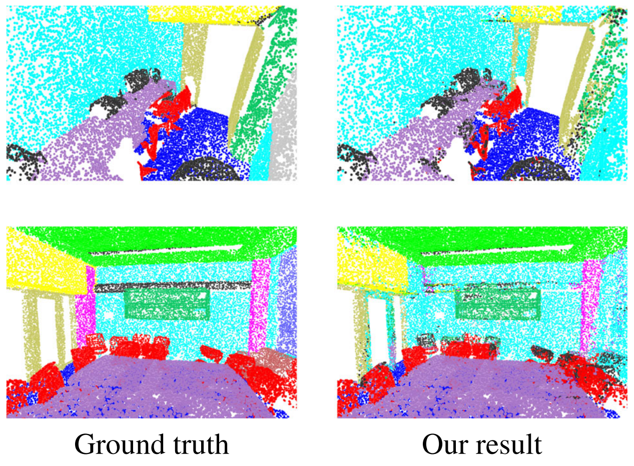

Fig. 7 Visual result of scene segmentation

Appendix C Additional visual results on shape part segmentation

In this section, we put more visual results of our method on the downstream shape part segmentation. We simply employ the Linear Classifier setting for this downstream task. Figure 8 shows the visual results of 32 models of 16 categories, involving 2 models per category. As we can see from the figure, with our unsupervised learned representations, a simple linear classifier for the downstream task can generate very similar visual results to the ground truth segmentation. It further confirms the effectiveness of our unsupervised method in learning distinguishable representations.

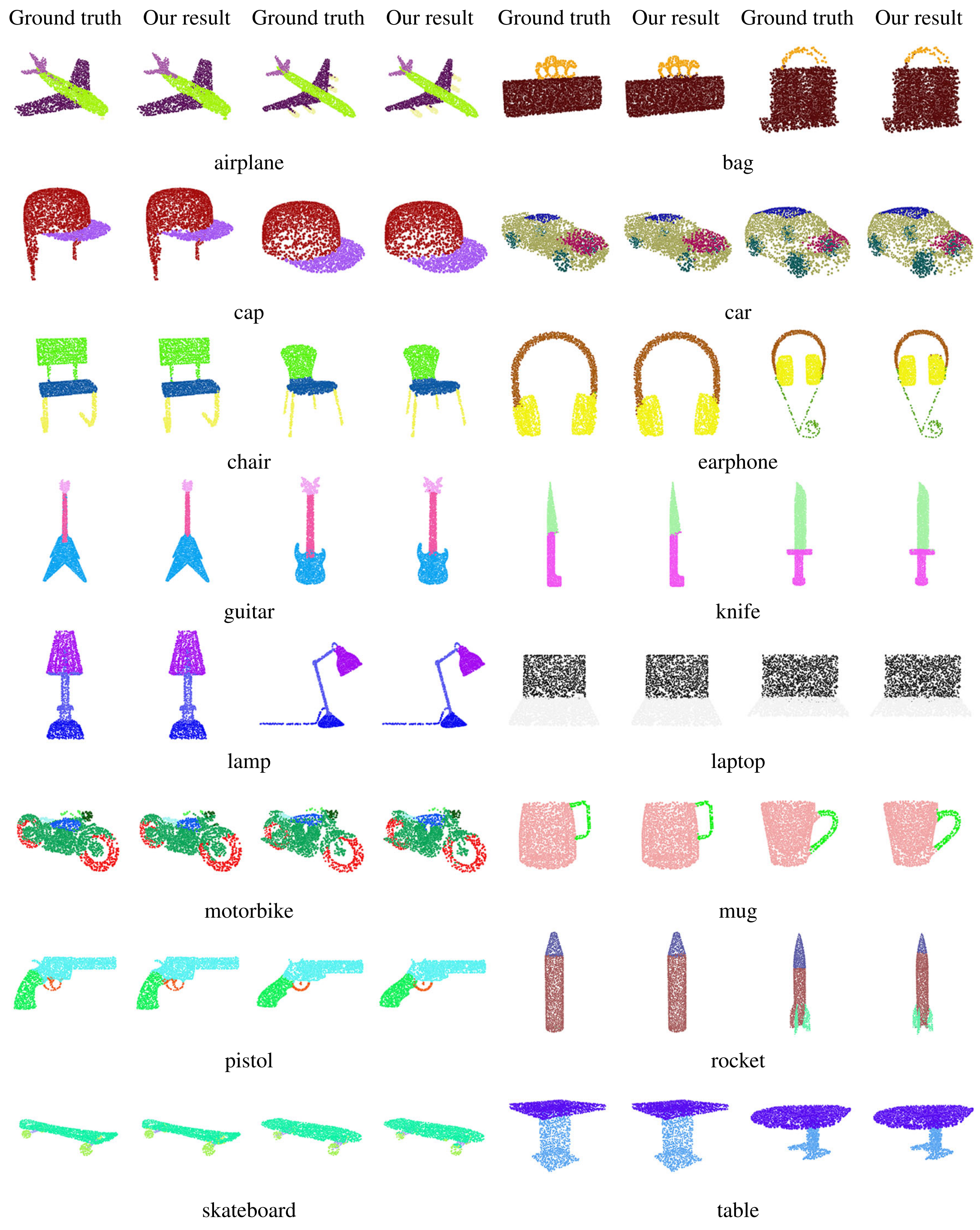

Fig. 8 Some examples of all 16 categories in ShapeNet Part dataset

References

1. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: a deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1912–1920 (2015)
2. Riegler, G., Osman Ulusoy, A., Geiger, A.: Octnet: learning deep 3d representations at high resolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3577–3586 (2017)
3. Wang, P.-S., Liu, Y., Guo, Y.-X., Sun, C.-Y., Tong, X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. ACM Trans. Graph. (TOG) **36**(4), 1–11 (2017)
4. Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E.: Multi-view convolutional neural networks for 3d shape recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 945–953 (2015)

5. Li, L., Zhu, S., Fu, H., Tan, P., Tai, C.-L.: End-to-end learning local multi-view descriptors for 3d point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1919–1928 (2020)
6. Lyu, Y., Huang, X., Zhang, Z.: Learning to segment 3d point clouds in 2d image space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12255–12264 (2020)
7. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: 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 (2017)
8. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems, pp. 5099–5108 (2017)
9. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: convolution on x-transformed points. In: Advances in Neural Information Processing Systems, pp. 820–830 (2018)
10. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International Conference on Machine Learning, pp. 40–49. PMLR (2018)
11. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 206–215 (2018)
12. Han, Z., Wang, X., Liu, Y.-S., Zwicker, M.: Multi-angle point cloud-vae: unsupervised feature learning for 3d point clouds from multiple angles by joint self-reconstruction and half-to-half prediction. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10441–10450. IEEE (2019)
13. Zhao, Y., Birdal, T., Deng, H., Tombari, F.: 3d point capsule networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1009–1018 (2019)
14. Zhang, D., Lu, X., Qin, H., He, Y.: Pointfilter: point cloud filtering via encoder-decoder modeling. IEEE Trans. Vis. Comput. Graph. (2020). https://doi.org/10.1109/TVCG.2020.3027069
15. Lu, D., Lu, X., Sun, Y., Wang, J.: Deep feature-preserving normal estimation for point cloud filtering. Comput. Aided Des. **125**, 102860 (2020). https://doi.org/10.1016/j.cad.2020.102860
16. Lu, X., Schaefer, S., Luo, J., Ma, L., He, Y.: Low rank matrix approximation for 3d geometry filtering. IEEE Trans. Vis. Comput. Graph. (2020). https://doi.org/10.1109/TVCG.2020.3026785
17. Lu, X., Wu, S., Chen, H., Yeung, S., Chen, W., Zwicker, M.: Gpf: Gmm-inspired feature-preserving point set filtering. IEEE Trans. Vis. Comput. Graph. **24**(8), 2315–2326 (2018). https://doi.org/10.1109/TVCG.2017.2725948
18. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.-H., Kautz, J.: Splatnet: sparse lattice networks for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2530–2539 (2018)
19. Zhou, H.-Y., Liu, A.-A., Nie, W.-Z., Nie, J.: Multi-view saliency guided deep neural network for 3-d object retrieval and classification. IEEE Trans. Multimed. **22**(6), 1496–1506 (2019)
20. Wu, W., Qi, Z., Fuxin, L.: Pointconv: deep convolutional networks on 3d point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9621–9630 (2019)
21. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y.: Spidercnn: Deep learning on point sets with parameterized convolutional filters. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 87–102 (2018)
22. Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8895–8904 (2019)
23. Komarichev, A., Zhong, Z., Hua, J.: A-cnn: annularly convolutional neural networks on point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7421–7430 (2019)
24. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. Acm Trans. Graph. (tog) **38**(5), 1–12 (2019)
25. Lin, Z.-H., Huang, S.-Y., Wang, Y.-C.F.: Convolution in the cloud: learning deformable kernels in 3d graph convolution networks for point cloud analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1800–1809 (2020)
26. Jiang, L., Shi, S., Tian, Z., Lai, X., Liu, S., Fu, C.-W., Jia, J.: Guided point contrastive learning for semi-supervised point cloud semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6423–6432 (2021)
27. Du, B., Gao, X., Hu, W., Li, X.: Self-contrastive learning with hard negative sampling for self-supervised point cloud learning. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 3133–3142 (2021)
28. Xu, C., Leng, B., Chen, B., Zhang, C., Zhou, X.: Learning discriminative and generative shape embeddings for three-dimensional shape retrieval. IEEE Trans. Multimed. **22**(9), 2234–2245 (2019)
29. Huang, J., Yan, W., Li, T.H., Liu, S., Li, G.: Learning the global descriptor for 3d object recognition based on multiple views decomposition. IEEE Trans. Multimed. (2020)
30. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3693–3702 (2017)
31. Wang, S., Suo, S., Ma, W.-C., Pokrovsky, A., Urtasun, R.: Deep parametric continuous convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2589–2597 (2018)
32. Li, J., Chen, B.M., Hee Lee, G.: So-net: self-organizing network for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9397–9406 (2018)
33. Zhao, H., Jiang, L., Fu, C.-W., Jia, J.: Pointweb: enhancing local neighborhood features for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5565–5573 (2019)
34. Xie, S., Liu, S., Chen, Z., Tu, Z.: Attentional shapecontextnet for point cloud recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4606–4615 (2018)
35. Fujiwara, K., Hashimoto, T.: Neural implicit embedding for point cloud analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11734–11743 (2020)
36. Yan, X., Zheng, C., Li, Z., Wang, S., Cui, S.: Pointasnl: robust point clouds processing using nonlocal neural networks with adaptive sampling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5589–5598 (2020)
37. Qiu, S., Anwar, S., Barnes, N.: Geometric back-projection network for point cloud classification. IEEE Trans. Multimed. (2021)
38. Chen, C., Qian, S., Fang, Q., Xu, C.: Hapgn: hierarchical attentive pooling graph network for point cloud segmentation. IEEE Trans. Multimed. (2020)
39. Liu, H., Guo, Y., Ma, Y., Lei, Y., Wen, G.: Semantic context encoding for accurate 3d point cloud segmentation. IEEE Trans. Multimed. (2020)
40. Rao, Y., Lu, J., Zhou, J.: Global-local bidirectional reasoning for unsupervised representation learning of 3d point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5376–5385 (2020)

41. Zhang, M., You, H., Kadam, P., Liu, S., Kuo, C.-C.J.: Pointhop: an explainable machine learning method for point cloud classification. IEEE Trans. Multimed. **22**(7), 1744–1755 (2020)
42. Xie, S., Gu, J., Guo, D., Qi, C.R., Guibas, L., Litany, O.: Pointcontrast: unsupervised pre-training for 3d point cloud understanding. In: European Conference on Computer Vision, pp. 574–591. Springer (2020)
43. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607. PMLR (2020)
44. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
45. Gadelha, M., Wang, R., Maji, S.: Multiresolution tree networks for 3d point cloud processing. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 103–118 (2018)
46. Han, Z., Shang, M., Liu, Y.-S., Zwicker, M.: View inter-prediction gan: unsupervised representation learning for 3d shapes by learning global shape memories to support local view predictions. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8376–8384 (2019)
47. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: an information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
48. Yi, L., Kim, V.G., Ceylan, D., Shen, I.-C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.: A scalable active framework for region annotation in 3d shape collections. ACM Trans. Graph. (ToG) **35**(6), 1–12 (2016)
49. Armeni, I., Sax, S., Zamir, A.R., Savarese, S.: Joint 2d-3d-semantic data for indoor scene understanding. arXiv preprint arXiv:1702.01105 (2017)
50. Klokov, R., Lempitsky, V.: 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 (2017)
51. Shen, Y., Feng, C., Yang, Y., Tian, D.: Mining point cloud local structures by kernel correlation and graph pooling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4548–4557 (2018)
52. Thomas, H., Qi, C.R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L.J.: Kpconv: flexible and deformable convolution for point clouds. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 6411–6420 (2019)
53. Atzmon, M., Maron, H., Lipman, Y.: Point convolutional neural networks by extension operators. ACM Trans. Graph. (TOG) **37**(4), 1–12 (2018)
54. Liu, X., Han, Z., Liu, Y.-S., Zwicker, M.: Point2sequence: learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8778–8785 (2019)
55. Hassani, K., Haley, M.: Unsupervised multi-task feature learning on point clouds. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 8160–8171 (2019)
56. Huang, Q., Wang, W., Neumann, U.: Recurrent slice networks for 3d segmentation of point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2626–2635 (2018)
57. Feng, Z., Zhou, Q., Gu, Q., Tan, X., Cheng, G., Lu, X., Shi, J., Ma, L.: Dmt: dynamic mutual training for semi-supervised learning. arXiv preprint arXiv:2004.08514 (2020)

Jincen Jiang received the B.E. degree from the College of Information Engineering, Northwest A&F University, China, in 2020. He is currently pursuing the Master degree with the College of Information Engineering, Northwest A&F University, China. His research interests include computer graphics, geometric modeling, and 3D deep learning.

Xuequan Lu is a Senior Lecturer at the School of Information Technology, Deakin University, Australia. He spent more than two years as a Research Fellow in Singapore. Prior to that, he earned his PhD at Zhejiang University (China) in June 2016. His research interests mainly fall into the category of visual computing, for example, 3D geometry modeling, processing and analysis, animation/simulation, 2D data processing and analysis. More information can be found at http://www.xuequanlu.com.

Wanli Ouyang received the Ph.D. degree from the Department of Electronic Engineering, Chinese University of Hong Kong. He is now a professor at Shanghai AI Lab. He was awarded Australian Research Council Future Fellowship. His research interests include AI4Science, computer vision, and pattern recognition.

Meili Wang is a professor at College of Information Engineering, Northwest A&F University. She received her Ph.D. degree in computer animation in 2011 at the National Centre for Computer Animation, Bournemouth University. Her research interests include computer graphics, geometric modeling, image processing, visualization and virtual reality.