Unsupervised 3D Learning for Shape Analysis via Multiresolution Instance Discrimination

Peng-Shuai Wang[1], **Yu-Qi Yang**[2,1], **Qian-Fang Zou**[3,1], **Zhirong Wu**[1], **Yang Liu**[1], **Xin Tong**[1]
[1]Microsoft Research Asia, [2]Tsinghua University, [3]University of Science and Technology of China

Abstract

We propose an unsupervised method for learning a generic and efficient shape encoding network for different shape analysis tasks. Our key idea is to jointly encode and learn shape and point features from unlabeled 3D point clouds. For this purpose, we adapt HRNet to octree-based convolutional neural networks for jointly encoding shape and point features with fused multiresolution subnetworks and design a simple-yet-efficient *Multiresolution Instance Discrimination* (MID) loss for jointly learning the shape and point features. Our network takes a 3D point cloud as input and output both shape and point features. After training, Our network is concatenated with simple task-specific back-ends and fine-tuned for different shape analysis tasks. We evaluate the efficacy and generality of our method with a set of shape analysis tasks, including shape classification, semantic shape segmentation, as well as shape registration tasks. With simple back-ends, our network demonstrates the best performance among all unsupervised methods and achieves competitive performance to supervised methods. For fine-grained shape segmentation on the PartNet dataset, our method even surpasses existing supervised methods by a large margin.

1 Introduction

3D shape analysis plays an important role in many graphics and vision applications. A key step in all shape analysis tasks is to extract representative features (or called descriptors) in different levels from 3D shapes. In particular, distinguishable shape instance features are preferred for shape classification, while per-point features are essential to fine-level analysis tasks, like semantic shape segmentation and registration.

Early methods compute handcrafted features of 3D shapes. Although these manually-designed features can preserve some good properties such as transformation invariant, they are difficult to be tailored to specific shape analysis applications. State-of-the-art methods integrate the feature extraction with specific shape analysis task and learn an end-to-end deep neural network with the supervision of labeled shape analysis results. The success of these supervised learning methods is built upon large-scale labeled datasets, and the networks optimized for one task are difficult to adapt to others.

Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Unsupervised pre-training methods first learn a feature extraction backbone network from an unlabeled dataset via carefully designed unsupervised pretext task losses. After that, the pre-trained backbone network is concatenated with task-specific back-end networks and refined for different downstream tasks via transfer learning. In computer vision and natural language processing tasks, unsupervised pre-training have demonstrated their advantages for reducing the workload of data labeling and network design (Wu et al. 2016; Yang et al. 2018; Deng, Birdal, and Ilic 2018; Zhao et al. 2019; Hassani and Haley 2019). However, these networks and training schemes cannot be easily adapted for 3D shape analysis due to irregular representation of 3D point clouds and multi-level shape features required by different shape analysis tasks. A set of unsupervised 3D learning methods (Wu et al. 2016; Yang et al. 2018; Deng, Birdal, and Ilic 2018; Zhao et al. 2019) have been proposed for extracting shape features from 3D point clouds, none of them offers a generic backbone network for different shape analysis tasks with competitive performance to the supervised methods.

In this paper, we present an unsupervised pre-training method for learning a generic 3D shape encoding network for 3D shape analysis. Our key observation is that a 3D shape is composed of its local parts and thus the feature for shape and points are coherent and should be encoded and trained jointly. Based on this observation, our shape encoding backbone network adapts HRNet (Wang et al. 2019a) to an octree-based convolutional network (Wang et al. 2017) for extracting and fusing features from both points and shapes via parallel multiresolution subnetworks and connections across subnetworks. It takes 3D point cloud as input and outputs an instance-wise feature of the whole 3D shape as well as point-wise features. Inspired by the instance discrimination designed for 2D image classification (Wu et al. 2018), we design a simple-yet-efficient Multi-resolution Instance Discrimination (MID) losses for supervision of extracted shape and point features, in which a shape instance discrimination loss classifies augmented copies of each shape instance of a 3D dataset in one class, while a point instance discrimination loss classifies the same points on the augmented copies of a shape instance in a class.

We trained our backbone shape encoding network (denoted as MID-Net) with ShapeNetCore55 (Chang et al. 2015) and evaluated its performance with simple back-ends in vari-

ous shape analysis tasks, including shape classification, two shape segmentation tasks, and 3D shape registration. Our experiments demonstrate that in all these tasks, our pre-trained backbone offers better performance than the same network trained with the labeled data of downstream tasks, especially as the amount of labeled data in the downstream tasks becomes small. Among all unsupervised 3D learning methods, our method achieves the best performance in all shape analysis tasks. Moreover, it achieves competitive performance to the state-of-the-art supervised methods in all tasks. In fine-grained PartNet segmentation, our method surpasses state-of-the-art supervised methods by a large margin.

2 Related Work

Supervised 3D feature learning. Discriminative shape features can be learned with the supervision of the labeled data in a task-specific manner. Supervised deep learning approaches often achieve the best performance on the datasets of shape classification and segmentation (Yi et al. 2017a; Mo et al. 2019; Yu et al. 2019). Existing 3D deep learning methods can be classified according to shape representations: multi-view based CNNs (Maturana and Scherer 2015; Su et al. 2015; Choy et al. 2016; Kalogerakis et al. 2017), volumetric and sparse-voxel-based CNNs (Wu et al. 2015; Graham 2015; Wang et al. 2017; Riegler, Ulusoy, and Geiger 2017; Graham, Engelcke, and van der Maaten 2018), point-based networks (Qi et al. 2017b,a; Li et al. 2018; Li, Chen, and Lee 2018), manifold-based CNNs (Boscaini et al. 2015, 2016; Hanocka et al. 2019) and graph-based approaches (Yi et al. 2017b; Wang et al. 2019b). Despite the good performance, the task-specific learned features are difficult to adapt to other tasks and preparing labeled 3D data is tedious and costly.

Unsupervised 3D feature learning. unsupervised learning methods aim to learn generic 3D shape representations from unlabeled data via carefully designed pretext tasks.

3D-GANs (Wu et al. 2016) train a generative adversarial network (GAN) on volumetric data, and its discriminator is used for extracting shape-level features. L-GANs learn deep shape representations by combining an autoencoder network and a GAN (Achlioptas et al. 2018). FoldingNets (Yang et al. 2018) and AtlasNets (Groueix et al. 2018) optimize an autoencoder by reconstructing shapes from deformed point clouds. Zhang and Zhu (2019) cascade two pretext tasks — part contrasting and object clustering to learn the shape feature space. The unsupervised shape-level features are mainly used for shape classification and retrieval.

Deng *et al*. (2018) and Shu *et al*. (2016) first extract handcrafted point-pair features from geometric patches, then uses an autoencoder to compress the features. SO-Nets (Li, Chen, and Lee 2018) extract hierarchical features from individual points and nodes of a self-organizing map. PointCapsNets (Zhao et al. 2019) extend capsule networks (Sabour, Frosst, and Hinton 2017) to 3D point clouds and are trained by the shape reconstruction loss. Hassani and Haley (2019) propose to train a multiscale graph-based encoder with multiple pretext tasks. Li *et al*. (2020) propose an unsupervised clustering task to learn point features for detecting distinctive shape regions. The work (Sauder and Sievers 2019) proposes a novel pretext task that reconstructs the voxel indices of randomly arranged points. Concurrently, Xie *et al*. (2020) use shape registration as the pretext task and adopts the point contrastive loss to learning point features; Yang *et al*. (2020) design a pretext task of finding correspondences between shapes belonging the same category based on the cycle-consistency.

Although the above works can generate both point-level and shape-level features, their pretext tasks do not impose explicit self-supervision on both levels. We fill this gap by discriminating different-level features simultaneously and achieve significant improvements.

Unsupervised pre-training. Unsupervised pre-training is actively studied due to its success in various fields. In the natural language processing field, BERT models (Devlin et al. 2018) use masked context prediction and next sentence prediction as the pretext tasks, and GPT models (Radford et al. 2018) prefer the language modeling task. In computer vision field, various pretext tasks or dedicated loss functions like colorization (Zhang, Isola, and Efros 2016), context prediction (Doersch, Gupta, and Efros 2015), motion segmentation (Pathak et al. 2017), iterative feature clustering (Caron et al. 2018), image instance discrimination (Wu et al. 2018) and contrastive losses (Hadsell, Chopra, and LeCun 2006; He et al. 2020), have also shown their strengths in learning effective image features.

Inspired by instance discrimination (Wu et al. 2018) which operates on a single image level, we propose to discriminate multiresolution instances of shapes and develop effective training schemes to reduce the computational and memory cost caused by the large number of instances which cannot be handled by Wu *et al*.'s approach (2018).

3 Method

Given a large unlabeled 3D shape collection that consists of N 3D models, we assume each 3D shape X_i is represented with a point cloud with M_i points, and denote the j-th point of X_i by $\mathbf{p}_{i,j}$. Here the term "unlabeled" means that the data has no shape category information or other human-annotated labels. Our goal is to train a shape encoder network f_θ that takes the point cloud of X_i as input and generate the representative and shape-level feature $\mathbf{s}_i$ and point-wise features $\{\mathbf{v}_{i,j}, j = 1, \ldots, M_i\}$ for X_i:

$$f_\theta : X_i \longmapsto [\mathbf{s}_i, \mathbf{v}_{i,1}, \cdots, \mathbf{v}_{i,M_i}], \tag{1}$$

where θ denotes the set of network parameters.

To train the above feature space in an unsupervised manner, we first augment shapes via various transformations and create multiresolution class labels for later training (see Section 3.1), then feed them into a deep neural network which maintains and fuses multi-scale resolution feature maps efficiently (see Section 3.2). We use the multi-resolution instance discrimination pretext task (see Section 3.3) to self-supervise the feature learning progress (see Section 3.4). Figure 1 shows the overview of our method.

3.1 Input data processing

Preprocessing and data augmentation. We pre-process the input point cloud to assign point normals via principal component analysis if the accurate normal information is not

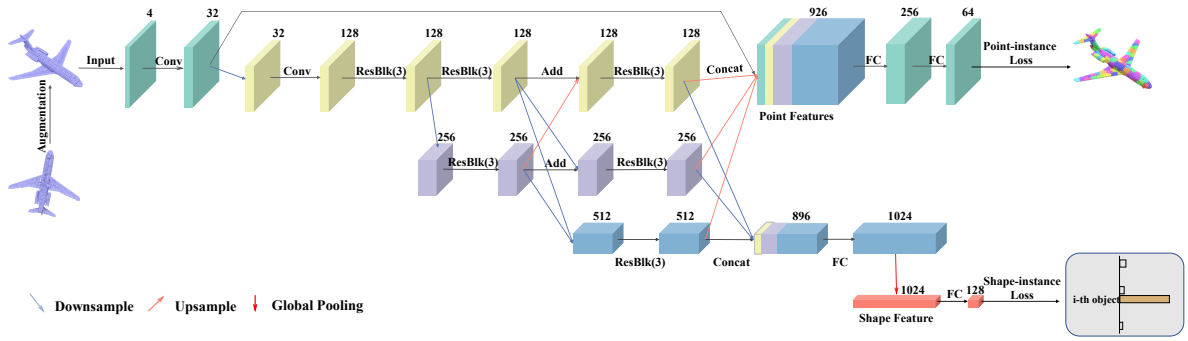

Figure 1: Overview of our multiresolution-instance-discrimination (MID) unsupervised pre-training pipeline. An augmented input point cloud via transformations is fed into the deep neural network which maintains and fuses multi-scale resolution feature maps. The shape-level features and point-wise features are extracted from the network, and they are encouraged to be discriminative and transformation-invariant under the supervision of the MID loss on both shape instance and point levels.

available. For data augmentation, we first normalize each point cloud inside a unit sphere, then generate shape instances with a transformation composed by random rotations, random translations within [-0.25, 0.25], and random scaling along each coordinate axis with the ratio within [0.75,1.25]. For the dataset in which the up-right directions of models are aligned, our random rotation is restricted on the rotation along the upright axis.

Multiresolution instance class creation. We label the transformed instances of the same shape with its index in the input dataset. Also, each point on the generated instance is labeled by the same index of the corresponding point in the input shape in the dataset. So in total, We create N shape-instance classes and M_i point-instance classes for each shape-instance class. These multiresolution class labels serve as self-supervision signals for our network training. As the total number of point-level classes of a large-scale shape dataset could be huge: $\sum_{i=1}^{N} M_i$, it will lead to huge memory consumption in network training. To overcome this issue, we introduce the concept of patch-instance class. On each shape X_i in the dataset, we over-segment it into K_i patches ($K_i \ll M_i$), and on each shape-instance class, we can create K_i patch-instance classes, similar to the construction of point-instance classes. In this way, the point-instance classes can be approximated by the patch-instance classes, which have a less total number. For simplicity, we use the K-Means algorithm to compute over-segmented patches and set $K_i = 100$.

3.2 Network design

We adopt HRNet (Wang et al. 2019a) to extract multiresolution instance features. Different from the conventional U-Net (Ronneberger, Fischer, and Brox 2015) architecture that cascades sequential convolutional layers from high to low resolutions and then recovers high resolution features from low-resolution ones in a reversed order, HRNet maintains parallel multiresolution subnetworks and simultaneously outputs multi-resolution features. The features extracted by subnetworks in different resolutions are fused in different intermediate stages of HRNet. For 3D shape encoding, HRNet can simultaneously output low-resolution shape-level features and high-resolution point-wise features with one network. This property well matches our observation about multiresolution instance features as introduced in Section 1. Also, we can easily apply loss functions for outputs in different resolutions and each loss function will contribute to the training of all subnetworks. We built HRNet upon an octree-based CNN framework (Wang et al. 2017) due to its efficiency in both computational cost and memory consumption and its natural multiresolution representation for building multiresolution subnetworks. A 3D point cloud is first converted to an octree representation, by default, in 64^3 resolution.

The details of the HRNet structure used in our method are illustrated in Figure 1. The numbers of feature channels are listed on the top of each feature map. The convolution kernel size is $3 \times 3 \times 3$. ResBlk(3) represents three cascaded ResNet blocks with a bottleneck structure (He et al. 2016). The Downsample operation is implemented by max-pooling, and the Upsample operation is the tri-linear up-sampling.

3.3 Multi-resolution instance discrimination

Our network training is supervised by two loss functions: shape-instance discrimination loss and point-instance discrimination loss. Both loss functions are based on input 3D models and their multiresolution instance class labels created in Section 3.1.

Shape-instance discrimination loss. The shape-level feature is supervised by a linear classifier that classifies shape instances into N classes, where each 3D shape X_i and its augmented instances are classified into the i-th class. The cross-entropy loss for the i-th class is defined as:

$$L_s(\mathbf{s}_i) = -\log \frac{\exp((\mathbf{s}_i \cdot \tilde{\mathbf{s}}_i)/\tau_s)}{\sum_{k=1}^{N} \exp((\mathbf{s}_i \cdot \tilde{\mathbf{s}}_k)/\tau_s)}, \qquad (2)$$

where $\tilde{\mathbf{s}}_i$ is the weight vector in the linear classifier for the i-th class. We follow the approach of (Wang et al. 2018; Wu et al. 2018) to normalize $\mathbf{s}_i$ and $\tilde{\mathbf{s}}_i$ to be unit-length and measure their difference by the cosine distance. τ_s is a parameter controlling the concentration level of the extracted features and is set to 0.1 empirically.

To obtain a good classification, an optimal $\tilde{\mathbf{s}}_i$ expects to be the average center of features of all shape instances in this class (Liu et al. 2018). By optimizing the loss function Equation (2), the features of shape instances under different transformations converge to $\tilde{\mathbf{s}}_i$ and thus they are invariant to the imposed transformations. Meanwhile, the loss function of Equation (2) tends to maximize the distance between $\mathbf{s}_i$ and other $\tilde{\mathbf{s}}_k (k \neq i)$, which makes the shape feature s_i of shape X_i is discriminative against the features of other 3D models.

Point-instance discrimination loss. To optimize point-wise features, we can also use a linear classifier to classify the points of a shape and its augmented copies into M_j classes, with the created supervision signals (Section 3.1). For points in the j-th point class, the cross-entropy loss is defined as:

$$L_p(\mathbf{v}_{i,j}) = -\log \frac{\exp((\mathbf{v}_{i,j} \cdot \tilde{\mathbf{v}}_{i,j})/\tau_p)}{\sum_{k=1}^{M_i} \exp((\mathbf{v}_{i,j} \cdot \tilde{\mathbf{v}}_{i,k})/\tau_p)}, \quad (3)$$

where $\mathbf{v}_{i,j}$ is the point-wise feature of j^{th} point of an augmented shape instance of X_i, $\tilde{\mathbf{v}}_{i,j}$ is the weight vector in the linear classifier for the j-th point class. All the point feature vectors are also unit-length and the control parameter τ_p is set to 0.1.

As discussed in Section 3.1, treating each point of a 3D shape as an individual class is impractical as it leads to a large number of classes for each object and results in huge memory consumption and computational cost for storing and updating the weights of linear classifiers. Therefore, we propose to approximate point-instance discrimination by patch-instance discrimination, and revise the loss function as:

$$L_p(\mathbf{v}_{i,j}) = -\log \frac{\exp((\mathbf{v}_{i,j} \cdot \tilde{\mathbf{v}}_{i,c(i,j)})/\tau_p)}{\sum_{c=1}^{K} \exp((\mathbf{v}_{i,j} \cdot \tilde{\mathbf{v}}_{i,c})/\tau_p)}, \quad (4)$$

where $c(i,j)$ is the index of the patch containing the j^{th} point on shape X_i.

MID Loss. By combining the above two multiresolution instance discrimination loss functions, we define the MID loss for an instance of shape X_i as:

$$L(X_i) = L_s(\mathbf{s}_i) + \sum\nolimits_{j=1}^{M_i} L_p(\mathbf{v}_{i,j})/M_i. \quad (5)$$

Note that although our method applies different linear classifiers for points of different 3D shapes, the shape encoder is shared by all 3D shapes.

3.4 Network training

Given the MID loss function defined above, we optimize the network parameters of f_θ with an input 3D shape collection by a stochastic gradient descent (SGD) algorithm. In each mini-batch, we randomly pick a set of 3D shapes from the dataset and generate an augmented shape for each original shape in runtime.

A naïve approach to train the network is to update the weights of the shape encoder network and the linear classifier together according to the back-propagated gradient of the MID loss function. However, we found that updating the linear classifiers in each mini-batch often makes the classifier training unstable and hinders the optimization of the shape encoder. Inspired by the temporal ensembling scheme in (Laine and Aila 2017), we update the weights of the linear shape-instance classifier slowly by

$$\tilde{\mathbf{s}}_i = (1 - \lambda_s) \cdot \tilde{\mathbf{s}}_i + \lambda_s \cdot \mathbf{s}_i, \quad (6)$$

where $\mathbf{s}_i$ is the shape-level feature of the current shape instance for the i-th shape-instance class, λ_s is a momentum parameter and is set to 0.5 in our implementation. For point-instance classifier, we use a similar update rule:

$$\tilde{\mathbf{v}}_{i,c} = (1 - \lambda_p) \cdot \tilde{\mathbf{v}}_{i,c} + \lambda_p \cdot \mathbf{v}_{i,c}, \quad (7)$$

where $\mathbf{v}_{i,c}$ is the average of the point-wise features of all points that belong to the patch class c, λ_p is a momentum parameter and is set to 0.5 too. A detailed training procedure is summarized in Algorithm 1.

4 MID-Nets for Shape Analysis

In this section, we first present our back-end network design and its training scheme and then discuss the performance of our method in each downstream task.

4.1 Back-end design and training

For each downstream task, we concatenate MID-Net with simple back-end layers. We use a one-layer fully-connected (FC) network for shape classification and two-layer FC for shape segmentation. We denote the MID-Net with the FC back-end as **MID-FC** and optimize the concatenated networks with two training schemes:

- **MID-FC(Fix)**: we fix the pre-trained MID-Net and only train the back-end with the labeled training data in each shape analysis task.
- **MID-FC(Finetune)**: we fine-tune both MID-Net and the FC back-end with the labeled training data in each shape analysis task. The MID-Net is initialized with the pre-trained weights, the FC back-end is randomly initialized.

To evaluate the impact of pre-training, we also randomly initialize both MID-Net and FC layers and train the network from scratch, denoted as **MID-FC(NoPre)**.

We trained our MID-Net on ShapeNet dataset (Chang et al. 2015) that consists of 57,449 3D shapes. The overall network parameter size is 1.5M, which is comparable to PointNet++ (Qi et al. 2017b) (1.4 M), DGCNN (Wang et al. 2019b) (1.5 M), and much smaller than PointCNN (Li et al. 2018) (8.2 M). The detailed training hyper-parameters are provided in the supplemental material.

Algorithm 1: Network training procedure

Input: A set of shapes $\{X_i\}_{i=1}^N$;
Output: Network f_θ, $\{\tilde{s}_i\}_{i=1}^N$ and $\{\{\tilde{v}_{i,c}\}_{c=1}^K\}_{i=1}^N$;
Assign multiresolution instance labels to $\{X_i\}_{i=1}^N$ according to Section 3.1
for *l in [0, max_iteration]* **do**
  Randomly sample a batch of B samples $\{X_{b_i}\}_{i=1}^B$;
  Compute loss $L = \frac{1}{B}\sum_{i=1}^B L(X_{b_i})$ (Equation (5));
  Compute gradient $\nabla L(\theta)$ and update θ with SGD;
  Update $\{\tilde{\mathbf{s}}_{b_i}\}_{i=1}^B$ and $\{\{\tilde{v}_{b_i,c}\}_{c=1}^K\}_{i=1}^B$ with Equation (6) & Equation (7).

Table 1: ModelNet40 shape classification. We list the performance of the state-of-the-art supervised methods (2nd column) and unsupervised methods (3rd column).

Our Method	Accu.	Supervised	Accu.	Unsupervised	Accu.
MID-FC(NoPre)	92.9	PointNet++	90.7	L-GAN	84.5
MID-FC(Fix)	90.3	PointCNN	92.2	FoldingNet	88.4
MID-FC(Finetune)	**93.1**	DGCNN	92.9	Multi-Task	89.1
		KPConv	92.9	PointCapsNets	89.3
		RS-CNN	92.9	ReconSpace	90.6

Table 2: The comparison of MID-FC(Fix) and FoldingNet for shape classification with a linear classifier as the back-end.

Data	1%	2%	5%	10%	20%
FoldingNet	56.4	66.9	75.6	81.2	83.6
MID-FC(Fix)	**61.5**	**73.1**	**80.2**	**84.2**	**86.9**
MID-FC(NoPre)	58.5	71.2	80.1	85.4	88.7
MID-FC(Finetune)	**68.1**	**77.2**	**83.6**	**88.4**	**90.2**

4.2 Shape classification

Dataset. For shape classification, we use the ModelNet40 benchmark (Wu et al. 2015), which contains 13,834 3D models across 40 categories: 9,843 models are used for training and 3,991 models for testing. We use the instance classification accuracy as the evaluation metric.

Results and comparisons. Table 1 lists the accuracy on the testing set of ModelNet40. For a fair comparison, we do not use the voting or orientation pooling strategy (Qi et al. 2017b; Li et al. 2018; Liu et al. 2019) to improve the results. As shown in the first column, our pre-trained MID-Net provides a good initialization to MID-FC(Finetune) and thus results in better accuracy (93.1%) than MID-FC(NoPre) (92.9%). Compared to the state-of-the-art supervised methods including PointNet++ (Qi et al. 2017b), PointCNN (Li et al. 2018), DGCNN (Wang et al. 2019b), KPConv (Thomas et al. 2019), and RS-CNN (Liu et al. 2019), our MID-FC(Finetune) also exhibits superior accuracy.

We also compare our method with other unsupervised methods including L-GAN (Achlioptas et al. 2018), FoldingNet (Yang et al. 2018), Multi-Task (Hassani and Haley 2019), PointCapsNet (Zhao et al. 2019), and ReconSpace (Sauder and Sievers 2019). Here all the unsupervised methods are pre-trained with the ShapeNet dataset and use one FC layer, *i.e.*, a linear classifier. Our MID-FC(Fix) achieves the second-best performance (90.3%) among all listed unsupervised approaches, and is slightly worse than ReconSpace (Sauder and Sievers 2019).

In Table 2, we evaluate the accuracy of MID-FC learned from different ratios of labeled data, where our method attains good performance and is better than FoldingNet (Yang et al. 2018). MID-FC(Finetune) further improves the classification accuracy and verifies the effectiveness of our pretraining.

4.3 Fine-grained PartNet segmentation

Dataset. We evaluate our method with fine-grained segmentation on the PartNet dataset (Mo et al. 2019), which is a challenging dataset that consists of 24,506 3D shapes in 17 categories with fine-grained (*i.e.*, Level-3) labels. Each shape contains 10,000 points and the part numbers in each shape category vary from 3 to 50. We follow the data split setup in (Mo et al. 2019). We use the mean IOU across all 17 categories as the evaluation metric. Note that the ShapeNet dataset used for our pre-training does not contain Door, Fridge, and Storage categories of PartNet.

Table 3: The semantic segmentation mIoU on PartNet.

Methods	mIoU	Methods	mIoU
MID-FC(NoPre)	58.4	PointNet++	49.6
MID-FC(Fix)	49.4	SpiderCNN	48.7
MID-FC(Finetune)	**60.8**	PointCNN	47.9

Table 4: The semantic segmentation mIoU on PartNet with different ratios of labeled data.

Model	1%	5%	10%	20%
MID-FC(NoPre)	14.5	27.7	29.5	41.6
MID-FC(Fix)	21.7	31.3	34.4	38.9
MID-FC(Finetune)	**21.8**	**35.4**	**40.2**	**46.3**

Results and comparisons. Table 3 shows the performance of MID-FC and other state-of-the-art supervised approaches, including PointNet++ (Qi et al. 2017b), SpiderCNN (Xu et al. 2018), and PointCNN (Li et al. 2018). For fair comparisons, we trained each model with the code provided by the original authors on the same dataset. And during the testing phase, we did not use the voting strategy to improve the result. Our MID-FC(Fix) outperforms most existing supervised approaches and is only slightly worse than PointNet++ by 0.2, which is significant considering the fact that MID-FC(Fix) only contains 2 trainable FC layers and optimizes less than 0.3M parameters. With fine-tuning, MID-FC(Finetune) achieves more improvements and the accuracy increases *11.2* points at least compared to other supervised approaches, and 2 point improvement over MID-FC(NoPre). The results clearly show the benefit of our pre-training. For the three categories that are only contained in PartNet while not in ShapeNet, MID-FC(Finetune) and MID-FC(Fix) still achieve better performance (see the supplemental material), which demonstrates the good generality of our pre-trained features.

The segmentation performance of all our three networks learned from different ratios of training data is reported in Table 4. MID-FC(Finetune) and MID-FC(Fix) have better performance than the supervised MID-FC(NoPre), and the accuracy of MID-FC(Fix) and MID-FC(Finetune) trained with 20% labeled data is comparable to other supervised methods trained with 100% labeled data.

4.4 ShapeNet Part segmentation

Dataset. We use the ShapeNet Part dataset (Yi et al. 2017a) which contains 16,881 3D point clouds, collected from 16 categories of ShapeNet. Each point cloud has 2 to 6 part labels. The number of points of a shape varies from 1000 to 3000. We follow (Yi et al. 2017a) to split the data. We use the mean IoU across all categories (C.mIoU) and the mean IoU across all instances (I.mIoU) as the evaluation metrics.

Results and Comparisons. Table 5 shows the IoU of our methods and other state-of-the-art methods, including super-

Table 5: The segmentation mean IoU across all categories (C.mIoU) and all instances (I.mIoU) on the ShapeNet Part.

Model (no voting)	C.mIOU	I.mIOU	Model (voting)	C.mIOU	I.mIOU
PointNet++	81.9	85.1	Submanifold	83.3	86.0
DGCNN	82.3	85.1	RS-CNN	84.0	86.2
SynSpecCNN	82.0	84.7	PointCNN	84.6	86.1
SPLATNet	83.7	85.4	KPConv	85.1	**86.4**
MID-FC(NoPre)	84.6	85.5	MID-FC(Finetune)	**85.2**	85.8
MID-FC(Fix)	83.4	84.6			
MID-FC(Finetune)	**84.8**	**85.5**			

Table 6: The segmentation mIoU on ShapeNet Part with different ratios of labeled data.

Model	C.mIoU		I.mIoU	
	1%	5%	1%	5%
PointCapsNets	-	-	67.0	70.0
Multi-Task	-	73.0	68.2	80.7
MID-FC(NoPre)	46.3	73.4	65.1	81.2
MID-FC(Fix)	59.7	77.9	72.4	81.1
MID-FC(Finetune)	**63.6**	**79.7**	**76.7**	**82.1**

vised methods such as PointNet++ (Qi et al. 2017b), SynSpecCNN (Yi et al. 2017b) and DGCNN (Wang et al. 2019b) and SPLATNet (Su et al. 2018), Submanifold (Graham, Engelcke, and van der Maaten 2018), KPConv (Thomas et al. 2019), PointCNN (Li et al. 2018), RS-CNN (Liu et al. 2019). On the left panel of Table 5, the methods did not use the voting strategy during the testing stage, and on the right panel, the methods used voting strategy during the testing stage. Compared with these supervised methods, MID-FC(Finetune) have a better performance. In Table 6, we also compare our method with fine-tuned unsupervised methods like Multi-Task (Hassani and Haley 2019) and PointCapsNet (Zhao et al. 2019) with 1% and 5% training data. Our MID-FC(Fix) and MID-FC(Finetune) also achieves better performance. And the improvements of our MID-FC(Finetune) over MID-FC(NoPre), especially when the training data is limited, clearly demonstrates the advantage of our pre-trained MID-Net.

4.5 Shape registration

The process of point cloud registration is to align a point cloud with its transformed version. To handle arbitrary rotations of shapes, we trained a new MID-Net with shapes augmented with arbitrary rotations. We regard the point-wise features extracted from our pre-trained MID backbone network as the point descriptors and use them to find closet point pairs from two input point clouds for computing the initial rigid transformation for the standard ICP algorithm.

Dataset. We conduct a comparison on the test set of ModelNet40 (Wu et al. 2015), which contains 2,468 man-made models across 40 categories. We normalize each shape inside a unit sphere and apply a random rigid transformation to the shape. In particular, the rotation angle along each coordinate axis is randomly sampled from $[0°, 360°]$, while the translation along each coordinate axis is randomly sampled from $[-0.25, 0.25]$. We evaluate the registration results by computing the Hausdorff distance between two registered shapes by different algorithms with the same ICP refinement.

Results and comparisons. We compare our method with four state-of-the-art methods, including Go-ICP (Yang et al. 2015), FGR (Zhou, Park, and Koltun 2016), PointLK (Aoki et al. 2019), and DCP (Wang and Solomon 2019). DCP and PointLK were originally trained with rotation angles sampled within $[0°, 45°]$. We re-trained their models with the code provided by the authors. However, we found that the trained networks perform poorly in dealing with large rotations. For all the methods except FGR which performs nonlinear optimization, we run the ICP algorithm to refine the initial registration. Figure 2 shows the accuracy curve within a given Hausdorff distance bound. The ground-truth curve and the curve by applying the standard ICP algorithm serve as the upper and lower bound of all the algorithms. Our method outperforms the other four methods.

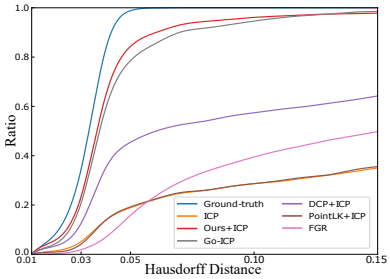

Figure 2: Point cloud registration results. The curve represents the ratio of registration results under a specific Hausdorff distance. Our method achieves the best performance.

5 Method Validation

In this section, we first evaluate the contributions of our octree-based HRNet and the MID loss, as well as the generality of our MID scheme for other network architectures. After that, we conduct a set of ablation studies to validate design decisions and parameter setting of our method.

5.1 Contributions of our network and MID

Contribution of our network. We compare the performance of MID-FC(NoPre) with the existing supervised methods that can achieve the best performance in each downstream task. Because the existing supervised methods and MID-FC(NoPre) have different network architectures but share similar supervised training schemes with random initializations, the difference between their performance can illustrate the contribution of our network.

As shown in Table 7, our octree-based HRNet achieves very similar performance to the best supervised methods in shape classification and ShapeNet-Part segmentation (with 0.0 and +0.5 differences, respectively). For fine-grained shape segmentation, the performance of MID-FC(NoPre) is significantly better (with +11.9 difference). The good performance on different downstream tasks illustrates that our octree-based HRNet offers an efficient network architecture for our generic pre-trained model.

Contribution of MID. For the contribution of the MID loss, we evaluate its contribution in each task by comparing the performance of MID-FC(NoPre) and MID-FC(Finetune). The

Table 7: Evaluation of our network and the MID scheme.

Model	Shape Classification	ShapeNet Segmentation	PartNet Segementation
Prior Art.	92.9 (DGCNN)	83.7 (SPLATNet)	47.9 (PointCNN)
MID-FC(NoPre)	92.9	84.2	58.4
MID-FC(Finetune)	93.1	84.8	60.8
PointNet-FC(NoPre)	90.1	81.9	43.8
PointNet-FC(Finetune)	90.1	83.4	49.4

Table 8: Ablation study of different design choices.

Model	Classification	Segementation	Model	Classification	Segementation
MID-FC	90.3	85.5	NoAug	82.3	83.9
MID-1Fusion	89.9	85.3	NoScale	89.2	84.3
MID-NoFusion	88.2	83.1	NoTrans	89.7	85.1
U-Net	89.3	84.5	NoRot	89.9	85.3
Point-Loss	89.5	85.3	50-Patch	90.1	84.9
Shape-Loss	88.9	83.3	200-Patch	90.0	85.5
			400-Patch	90.2	85.4

two models share the same network architecture but are trained with different learning schemes. The MID-FC(NoPre) is trained from scratch with the task-specific labeled data by a supervised scheme, while the MID-FC(Finetune) is initialized with the weights of MID-Net pre-trained via our MID scheme and then refined with the task-specific labeled data.

As shown in Table 7, the performance gain of MID-FC(Finetune) over the supervised MID-FC(NoPre) counterparts in all three tasks (+0.2, +0.6, +2.4, respectively) clearly demonstrates the contribution of our MID scheme for different downstream tasks.

Generality of the MID scheme. To validate the generality and advantage of the MID scheme for pre-training, we apply the MID scheme to PointNet++ with the default network in (Qi et al. 2017b). Similar to MID-Net, we concatenate PointNet++ with a two-layered FC layer and fine-tune the networks (denoted as PointNet-FC(Finetune)) for all the tasks. For comparison, we also train the same concatenated PointNet++ networks from scratch with the task-specific training data (denote as PointNet-FC(NoPre)).

Table 7 shows that the performance gain between PointNet-FC(Finetune) and PointNet-FC(NoPre) in all three tasks exhibits the efficiency of our MID pre-training scheme to other network architectures, and the inferior performance of PointNet-FC(Finetune) to MID-FC(Finetune) also illustrates the importance of network architecture to the pre-trained model and the advantage of our octree-based HRNet.

5.2 Ablation Study

We pre-train our networks with one modified component and keep all other settings unchanged. After pre-training, we test the modified models with shape classification and ShapeNet-Part segmentation tasks described in the last section with MID-FC(Fix). To better evaluate the efficacy of our pre-training in ablation studies, we use a denser version of point clouds provided by (Wang et al. 2017) for the ShapeNet-Part segmentation task which has a similar point density to the data used in the pre-training.

HRNet versus other network structures. Compared with the U-Net (Ronneberger, Fischer, and Brox 2015), the key advantages of HRNet (Wang et al. 2019a) are the parallel multi-resolution subnetworks and feature fusion in different resolutions. We develop two alternative networks that gradually remove the fusion layers between subnetworks. The MID-1Fusion removes the first fusion layer from HRNet, while the MID-NoFusion removes all fusion layers in HRNet.

As shown in Table 8, the accuracy of the network decreases as we drop more fusion layers (*e.g.* , decrease 0.4% for MID-1Fusion and 2.1% for MID-NoFusion in shape classification), which clearly demonstrates the importance of the feature fusion. And the accuracy of U-Net drops 1.4% in shape classification and 0.5% in shape segmentation, which validates the advantage of HRNet in 3D pre-training.

MID Loss versus single-level Loss. To demonstrate the advantages of multi-resolution instance loss functions, we train two networks, each of which is trained with one loss function only. Compared with the MID-Net trained with two loss functions, the performance of these two networks (denoted as Shape-Loss and Point-Loss in Table 8) drops, which indicates that each loss function makes its own contribution to the full network training and affects the performance of shape encoding in both shape and point levels.

And when the training data is limited, the performance gap is further enlarged: with only 1% of the training data, the testing accuracy on the classification task and category IoU on the segmentation task of our network trained with two loss functions are 61.8 and 74.6, respectively; with only the shape loss, the accuracy and IoU drop to 60.5 and 61.6; with only the point loss, the accuracy and IoU drop to 54.4 and 72.2.

Augmentation scheme. To study the impact of different augmentation schemes, we train the network with three different augmentation schemes, each of which drops one kind of transformation from the original augmentation scheme, respectively. Compared with the MID-Net trained with full augmentation, the performances of three networks trained with new augmentation schemes (denoted as NoRot, NoScale, NoTrans, and NoAug) shown in Table 8 decrease.

Patch number. To test the impact of the patch number, we train the networks with different K. As shown in Table 8, as the number of clusters increases from 100 to 400, the performance of the resulting networks increases less than 0.2% in both shape segmentation and classification tests. We thus set K as 100 in our current implementation to achieve a good balance between training cost and model performance.

6 Conclusion

We propose an unsupervised pre-training method for learning a generic backbone network from unlabeled 3D shape collections for shape analysis. To this end, we design an octree-based HRNet as backbone network architecture and a simple-yet-efficient MID loss for pre-training. The ablation study validates the advantages of joint shape and point feature encoding and training enabled by our design in the unsupervised pre-training and downstream shape analysis tasks. Our MID-Net offers state-of-the-art performance for various downstream shape analysis tasks, especially for tasks with small training sets.

References

Achlioptas, P.; Diamanti, O.; Mitliagkas, I.; and Guibas, L. 2018. Learning representations and generative models for 3D point clouds. In *ICLR*.

Aoki, Y.; Goforth, H.; Arun Srivatsan, R.; and Lucey, S. 2019. PointNetLK: Robust & Efficient Point Cloud Registration Using PointNet. In *CVPR*.

Boscaini, D.; Masci, J.; Melzi, S.; Bronstein, M. M.; Castellani, U.; and Vandergheynst, P. 2015. Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. *Comput. Graph. Forum* 34(5).

Boscaini, D.; Masci, J.; Rodolà, E.; and Bronstein, M. M. 2016. Learning shape correspondence with anisotropic convolutional neural networks. In *NeurIPS*.

Caron, M.; Bojanowski, P.; Joulin, A.; and Douze, M. 2018. Deep clustering for unsupervised learning of visual features. In *ECCV*.

Chang, A. X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; Xiao, J.; Yi, L.; and Yu, F. 2015. ShapeNet: An information-rich 3D model repository. Technical Report arXiv:1512.03012 [cs.GR].

Choy, C. B.; Xu, D.; Gwak, J.; Chen, K.; and Savarese, S. 2016. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *ECCV*.

Deng, H.; Birdal, T.; and Ilic, S. 2018. PPF-FoldNet: Unsupervised learning of rotation invariant 3D local descriptors. In *ECCV*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* .

Doersch, C.; Gupta, A.; and Efros, A. A. 2015. Unsupervised visual representation learning by context prediction. In *ICCV*.

Graham, B. 2015. Sparse 3D convolutional neural networks. In *BMVC*.

Graham, B.; Engelcke, M.; and van der Maaten, L. 2018. 3D semantic segmentation with submanifold sparse convolutional networks. In *CVPR*.

Groueix, T.; Fisher, M.; Kim, V. G.; Russell, B. C.; and Aubry, M. 2018. AtlasNet: A Papier-Mâché approach to learning 3D surface generation. In *CVPR*.

Hadsell, R.; Chopra, S.; and LeCun, Y. 2006. Dimensionality reduction by learning an invariant mapping. In *CVPR*.

Hanocka, R.; Hertz, A.; Fish, N.; Giryes, R.; Fleishman, S.; and Cohen-Or, D. 2019. MeshCNN: A network with an edge. *ACM Trans. Graph. (SIGGRAPH)* 38(4).

Hassani, K.; and Haley, M. 2019. Unsupervised multi-task feature learning on point clouds. In *ICCV*.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *CVPR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Kalogerakis, E.; Averkiou, M.; Maji, S.; and Chaudhuri, S. 2017. 3D shape segmentation with projective convolutional networks. In *CVPR*.

Laine, S.; and Aila, T. 2017. Temporal ensembling for semi-supervised learning. *ICLR* .

Li, J.; Chen, B. M.; and Lee, G. H. 2018. SO-Net: Self-organizing network for point cloud analysis. In *CVPR*.

Li, X.; Yu, L.; Fu, C.-W.; Cohen-Or, D.; and Heng, P.-A. 2020. Unsupervised detection of distinctive regions on 3D shapes. *ACM Trans. Graph.* .

Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; and Chen, B. 2018. PointCNN: Convolution on X-transformed points. In *NeurIPS*.

Liu, Y.; Fan, B.; Xiang, S.; and Pan, C. 2019. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*.

Liu, Y.; Song, G.; Shao, J.; Jin, X.; and Wang, X. 2018. Transductive centroid projection for semi-supervised large-scale recognition. In *ECCV*.

Maturana, D.; and Scherer, S. 2015. VoxNet: A 3D convolutional neural network for real-time object recognition. In *International Conference on Intelligent Robots and Systems (IROS)*.

Mo, K.; Zhu, S.; Chang, A. X.; Yi, L.; Tripathi, S.; Guibas, L. J.; and Su, H. 2019. PartNet: A Large-scale Benchmark for Fine-grained and Hierarchical Part-level 3D Object Understanding. In *CVPR*.

Pathak, D.; Girshick, R.; Dollár, P.; Darrell, T.; and Hariharan, B. 2017. Learning features by watching objects move. In *CVPR*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

Riegler, G.; Ulusoy, A. O.; and Geiger, A. 2017. OctNet: Learning deep 3D representations at high resolutions. In *CVPR*.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*.

Sabour, S.; Frosst, N.; and Hinton, G. E. 2017. Dynamic routing between capsules. In *NeurIPS*.

Sauder, J.; and Sievers, B. 2019. Self-supervised deep learning on point clouds by reconstructing space. In *NeurIPS*.

Shu, Z.; Qi, C.; Xin, S.; Hu, C.; Wang, L.; Zhang, Y.; and Liu, L. 2016. Unsupervised 3D shape segmentation and co-segmentation via deep learning. *Comput. Aided Geom. Des.* 43.

Su, H.; Jampani, V.; Sun, D.; Maji, S.; Kalogerakis, E.; Yang, M.-H.; and Kautz, J. 2018. SPLATNet: Sparse lattice networks for point cloud processing. In *CVPR*.

Su, H.; Maji, S.; Kalogerakis, E.; and Learned-Miller, E. 2015. Multi-view convolutional neural networks for 3D shape recognition. In *ICCV*.

Thomas, H.; Qi, C. R.; Deschaud, J.-E.; Marcotegui, B.; Goulette, F.; and Guibas, L. J. 2019. KPConv: Flexible and deformable convolution for point clouds. In *ICCV*.

Wang, H.; Wang, Y.; Zhou, Z.; Ji, X.; Gong, D.; Zhou, J.; Li, Z.; and Liu, W. 2018. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*.

Wang, J.; Sun, K.; Cheng, T.; Jiang, B.; Deng, C.; Zhao, Y.; Liu, D.; Mu, Y.; Tan, M.; Wang, X.; Liu, W.; and Xiao, B. 2019a. Deep high-resolution representation learning for visual recognition. arXiv preprint arXiv:1908.07919.

Wang, P.-S.; Liu, Y.; Guo, Y.-X.; Sun, C.-Y.; and Tong, X. 2017. O-CNN: Octree-based convolutional neural networks for 3D shape analysis. *ACM Trans. Graph. (SIGGRAPH)* 36(4).

Wang, Y.; and Solomon, J. M. 2019. Deep Closest Point: Learning Representations for Point Cloud Registration. In *ICCV*.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019b. Dynamic graph CNN for learning on point clouds. *ACM Trans. Graph.* 38(5).

Wu, J.; Zhang, C.; Xue, T.; Freeman, W. T.; and Tenenbaum, J. B. 2016. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *NeurIPS*.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3D ShapeNets: A deep representation for volumetric shape modeling. In *CVPR*.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *CVPR*.

Xie, S.; Gu, J.; Guo, D.; Qi, C. R.; Guibas, L. J.; and Litany, O. 2020. PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding. *ECCV* .

Xu, Y.; Fan, T.; Xu, M.; Zeng, L.; and Qiao, Y. 2018. SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters. In *ECCV*.

Yang, J.; Li, H.; Campbell, D.; and Jia, Y. 2015. Go-ICP: A globally optimal solution to 3D ICP point-set registration. *IEEE Trans. Pattern Anal. Mach. Intell.* 38(11).

Yang, L.; Liu, W.; Cui, Z.; Chen, N.; and Wang, W. 2020. Mapping in a cycle: Sinkhorn regularized unsupervised learning for point cloud shapes. *ECCV* .

Yang, Y.; Feng, C.; Shen, Y.; and Tian, D. 2018. FoldingNet: Point cloud auto-encoder via deep grid deformation. In *CVPR*.

Yi, L.; Shao, L.; Savva, M.; and etal. 2017a. Large-scale 3D shape reconstruction and segmentation from ShapeNet Core55. arXiv:1710.06104 [cs.CV].

Yi, L.; Su, H.; Guo, X.; and Guibas, L. 2017b. SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation. In *CVPR*.

Yu, F.; Liu, K.; Zhang, Y.; Zhu, C.; and Xu, K. 2019. PartNet: A Recursive Part Decomposition Network for Fine-grained and Hierarchical Shape Segmentation. In *CVPR*.

Zhang, L.; and Zhu, Z. 2019. Unsupervised feature learning for point cloud understanding by contrasting and clustering using graph convolutional neural networks. In *3DV*.

Zhang, R.; Isola, P.; and Efros, A. A. 2016. Colorful image colorization. In *ECCV*.

Zhao, Y.; Birdal, T.; Deng, H.; and Tombari, F. 2019. 3D point capsule networks. In *CVPR*.

Zhou, Q.-Y.; Park, J.; and Koltun, V. 2016. Fast global registration. In *ECCV*.