Learning by Restoring Broken 3D Geometry

Jinxian Liu, Bingbing Ni, Ye Chen, Zhenbo Yu, and Hang Wang

***Abstract*—The key point for an experienced craftsman to repair broken objects effectively is that he must know about them deeply. Similarly, we believe that a model can capture rich geometry information from a shape/scene and generate discriminative representations if it is able to find distorted parts of shapes/scenes and restore them. Inspired by this observation, we propose a novel self-supervised 3D learning paradigm named learning by restoring broken shapes/scenes (collectively called 3D geometry). We first develop a destroy-method cluster, from which we sample methods to break some local parts of an object. Then the destroyed object and the normal object are both sent into a point cloud network to get representations, which are employed to segment points that belong to distorted parts and further reconstruct/restore them to normal. To perform better in these two associated pretext tasks, the model is constrained to capture useful object features, such as rich geometric and contextual information. The object representations learned by this self-supervised paradigm transfer well to different datasets and perform well on downstream classification, segmentation and detection tasks. Experimental results on shape datasets and scene datasets demonstrate that our method achieves state-of-the-art performance among unsupervised methods. We also show experimentally that pre-training with our framework significantly boosts the performance of supervised models.**

***Index Terms*—3D point cloud, shape, scene, self-supervised, representation.**

I. Introduction

WE LIVE in a 3D world. It is natural to do shape and scene understand in 3D domain. In the meanwhile, the growing of low-cost 3D sensors, e.g., LiDAR and RGB-D cameras, makes collecting 3D data easier and 3D understanding a tremendous demand recently. Supervised learning is a mainstay for large discriminative models in 3D computer vision [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], while large amounts of human-annotated data are the key to achieving state-of-the-art performance. However, labeling such kinds of irregular 3D data would be extremely labor-intensive. This directly results in the small-scale of existing 3D datasets and limits the performance of models on complex scenes.

In 2D domain, steady progress in un-/self-supervised representation learning [15], [16], [17], [18], [19], [20], [21] has emerged with encouraging results on multiple 2D visual tasks. A Large-scale unlabeled dataset is fully utilized to pre-train a model with elaborately designed learning schemes, and the model could be transferred to a usually much smaller target set and help boost its performance once fine-tuned. Contrastive learning [20], [21], [22] is the most successful paradigm among these methods in 2D domain. This motivates us that un-/self-supervised learning opens up the possibility to utilize practically infinite unlabeled data. Compared with 2D images, we hold a point that 3D data naturally contains richer structural and geometric information, which could be fully extracted in a self-supervised manner and further utilized to transfer to a labeled dataset. Moreover, 3D tasks are potentially the biggest beneficiaries of self-supervised learning due to that it is significantly harder to annotate 3D data than 2D counterpart. In general, we firmly believe that it is necessary and appropriate to explore un-/self-supervised learning in 3D domain.

Recently, some works [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34] pay more attention to self-supervised learning (SSL) on 3D domain. Most of these works explore SSL in a generative or context-instance contrastive paradigm. [23], [24], [25], [26] construct generative models to learn representations of point clouds without manually-annotated labels. However, point representations learned by these methods are not strong enough to be used for complex downstream tasks such as segmentation and detection. CAD shape retrieval and classification are their application scenarios, yet they do not perform well. [26], [27], [28], [30] develop various context-instance/local-global contrastive self-supervised methods to learn more discriminative representations. However, the pretext tasks proposed in these methods are usually easy to converge so that they get trivial solutions according to some clues that have nothing to do with shape geometry. Moreover, most of these methods are only applied to shape point clouds and it is difficult to transfer them to real scene samples. [31], [32], [33] are works published recently that follow most popular paradigm in 2D domain, i.e., instance-instance contrastive. [31] takes different views of a point cloud as positive samples, while [33] takes single view 3D depth scans as input and merge results of voxel and point data to enhance its performance. Methods following the instance-instance contrastive paradigm do achieve great results on classification tasks. However, elaborately sampling hard examples and designing effective metrics are tedious for achieving stable and good results. Moreover, lacking modeling of local parts limits them to perform better on some fine-grained tasks such as detection and segmentation.

Hence, developing a simple and general self-supervised framework for both 3D shapes and scenes without elaborately constructing contrast pairs and designing learning metrics is of the utmost urgency. To this end, we propose a self-supervised

Manuscript received 15 March 2022; revised 3 March 2023; accepted 26 March 2023. Date of publication 3 April 2023; date of current version 4 August 2023. This work was supported in part by the National Science Foundation of China under Grants U20B2072 and 61976137 and in part by SJTU Medical Engineering Cross Research under Grant YG2021ZD18. Recommended for acceptance by J. M. Solomon. *(Corresponding author: Bingbing Ni.)*

The authors are with the Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: liujinxian@sjtu.edu.cn; nibingbing@sjtu.edu.cn; chenye123@sjtu.edu.cn; yuzhenbo@sjtu.edu.cn; wang–hang@sjtu.edu.cn).

Digital Object Identifier 10.1109/TPAMI.2023.3263867

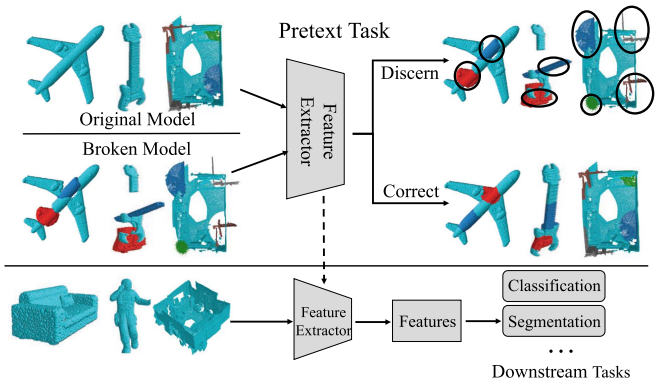

Fig. 1. Illustration of our main idea. As shown, we destroy the shape/scene parts with certain heuristic methods and there is a huge mismatch between the distorted parts and the normal parts. We can easily distinguish the distorted parts because we know the geometric characteristics of the object. Hence we think a strong representation that encodes effective structure information should also have the ability. We destroy shape/scene parts and train a network to distinguish the destroyed parts and restore them to normal in an unsupervised pretext task. Success in the pretext task indicates the network captures strong shape representations, which can transfer well to downstream tasks.

learning scheme for point clouds in a context-instance paradigm in this work. Our defined pretext task is intuitive and the model performs better than all methods mentioned above on downstream tasks. The proposed method is based on the principle that each object in the 3D world is made up of shape parts/elements/primitives (they do not need to contain specific semantic information). The rule how these parts match with each other and form the whole object reflects the robust geometric characteristics, which imply local structure information and semantic knowledge of the object. Hence, distorted local parts will break such geometric characteristics of the shape. One can easily distinguish distorted parts and even restore them if he fully captures the geometric information of such an object. Motivated by this principle, we think that a good 3D representation that encodes rich structural and semantic information should have the ability to find broken parts and correct them.

Inspired by such observations, we propose a novel self-supervised learning framework, i.e., learning by restoring broken 3D geometry, to capture representations of unlabeled 3D point clouds. Theoretically, our proposed scheme works for any type of 3D data (e.g., point clouds, voxel, mesh) with appropriate adjustments. However, we take point cloud as our input for its simplicity and broad applicability. The overview of our framework is shown in Fig. 1, which is formulated as destroying local parts of a 3D point cloud sample and encouraging the network to distinguish the destroyed shape/scene parts and restore them to normal. For simplicity, we collectively refer to shapes and scenes as objects later in this paper. For success in this pretext task, the trained model is constrained to capture rich geometric and structural information of the 3D point cloud. To this end, we first design an object-destroying module to break some local parts of the 3D samples. Specifically, we randomly sample some points of a point cloud, and then aggregate their neighbors to form some local parts. To destroy these sampled local parts, we develop a destroy-method cluster, from where we sample various methods to destroy local parts of the 3D sample. In the destroy-method cluster, we add some common data augmentation methods, such as random rotation, random scaling, *etc*. In addition, we design some simple yet effective parts destroying methods to enrich this cluster. For example, we exchange sampled local parts, replace the local parts with standard spheres or drag the local parts to deform them. Given a normal 3D point cloud, we randomly sample 1-3 methods to destroy its parts. Then we pair the normal sample and destroyed sample and send them into a shared network to extract features respectively, which are concatenated to perform destroyed parts searching and restoring. Note that our proposed framework is agnostic of network and can be applied to different point cloud networks like PointNet [1], PointNet++ [2], RSCNN [11] and MinkowskiNet [35]. In this paper, we modify PointNet, RSCNN and MinkowskiNet as our feature extractors respectively to evaluate our proposed method. With the features of the normal shape, accurate structure information is obtained so that the network is capable of performing better on the pretext tasks. To perform destroyed parts searching and restoring when given the concatenated features, we design two branches that can be summarized as: 1) Distinguishing Branch: we implement a point-wise classifier to segment points that belong to the distorted parts; 2) Restoring Branch: we further design a self-reconstruction module to correct the distorted shape based on the outputs of the Distinguishing branch.

After training the model on our proposed pretext task, we can use the model to perform various downstream tasks, such as classification, segmentation and detection. In this work, we follow two main settings to conduct experiments on shape datasets and real large scene datasets respectively. For setting 1, we utilize the ShapeNet [36] dataset as our source set for self-supervised pre-training and evaluate the learned features on two important 3D understanding tasks, i.e., shape classification and segmentation. Experimental results on several datasets indicate that our method achieves state-of-the-art performance among unsupervised models on both classification and segmentation tasks. We also show experimentally that pre-training with our framework significantly boosts the performance of supervised models. On the segmentation task, we also explore the effectiveness of the learned features in a semi-supervised setting and our method outperforms previous methods [31], [37], especially when labels are most limited. In addition, our per-trained model achieves competitive results on downstream tasks when only using PointNet as the backbone network, which demonstrates the strong feature learning ability of our framework. For setting 2, we take MinkowskiNet as the backbone and train the model with proposed pretext tasks on ScanNet [38]. Then we perform downstream tasks (i.e., scene segmentation and object detection) on S3DIS [39], SUN RGB-D [40] and ScanNet following a pretraining-and-finetuning paradigm. Experimental results show that our method performs better on most tasks than published works.

Our main contributions are summarized as follows:

1) We propose a novel self-supervised learning scheme i.e., learning by restoring broken 3D geometry. This new learning framework is general for shape and scene data.
2) We design a destroy-method cluster to break local parts of a 3D point cloud sample. Various methods can be added to this cluster to further improve its performance.

3) To make a fair comparison with various methods, we follow two main settings and perform large amounts of experiments on shape datasets and scene datasets.
4) Experimental results show that our method performs better than state-of-the-art methods and shows great generalization ability.

II. Related Work

Self-Supervised Learning on 2D Domain. To overcome the challenge that annotating data is high cost, many unsupervised learning methods [15], [16], [17], [18], [19], [20], [21], [22], [41], [42], [43], [44], [45] are proposed. Self-supervised learning is a kind of unsupervised learning method. It requires the network to find free and plentiful supervised signals for training. There are many self-supervised learning methods that have been proposed in 2D vision (image/video processing). In [41], authors propose to learn representations of images by letting the network solve the Jigsaw puzzle, where random pairs of patches from an unlabeled image are extracted and sent into a siamese network to predict position relationship between them. Furthermore, [42] defines a sort problem to perform self-supervised learning, where serval patches are extracted from an image regularly and a network is utilized to sort randomly shuffled input patches. And many other works are proposed such as colorization [43], inpainting [15], predicting rotation angle [44], *etc*. In addition, many works [20], [21], [22], [41], [44], [45] focus on contrastive self-supervised learning recently and achieve superior performance. They shed light on the potential of discriminative models for representation, and can be divided into two types: context-instance contrast and instance-instance contrast. The context-instance contrast is also called global-local contrast, which focuses on modeling the belonging relationship between the local feature and its global representation. [41] and [44] can be incorporated into such kind of contrast. Instance-instance contrastive learning directly explores the relationships between different samples with instance-level representations as metric learning does. CMC [45], MoCo [20], SimCLR [21], and BYOL [22] are recently proposed self-supervised algorithms based on instance-instance contrast, and they outperform the context-instance contrastive methods. Although these methods achieve superior performance on the classification task, they do not perform very well on pixel-wise tasks such as segmentation due to that they pay no attention to the local and fine-grained context information. In the field of video understanding, [46] explores to design a self-supervised learning scheme of finding corresponding pairs using visual tracking. The temporal order of a video [47] is also utilized as supervised signals for learning video representations. These proposed self-supervised learning methods are all designed for 2D data (image and video). Self-supervised learning, especially for that following instance-instance contrast paradigm, has achieved great success in 2D domain. However, only few works explore the effectiveness of self-supervised learning on 3D domain. In this work, we propose a novel self-supervised learning scheme for 3D data (point cloud) based on context-instance contrast paradigm, which achieves state-of-the-art performance on various downstream tasks.

Deep Learning on Point Cloud Understanding. PointNet [1] is a pioneering work to directly consume unordered and unstructured 3D point clouds, where MLPs and global max-pooling are utilized to obtain both point-wise features and global structure information. Despite PointNet well handles order invariances of input data and achieves strong performance, it fails to aggregate point-wise embeddings and capture local contextual information among points. PointNet++ [2] mitigates this issue by proposing a hierarchical learning architecture, where multi-scale local point embeddings are grouped. Several subsequent works [3], [4], [6], [48], [49], [50], [51], [52], [53], [54], [55], [56] employ neighbor searching to aggregate the representations of neighbor points and capture local geometry information. In addition, some works [57], [58] imitate CNN to design regular kernels and conduct convolution based on the correlation between kernels and the local point set. Furthermore, a hierarchical framework is usually constructed by stacking elaborately designed convolution layers. Attention and transformer are also introduced into the point cloud learning frameworks. Recently, some works [59], [60], [61] pay more attention to designing efficient point cloud learning architectures to facilitate real-time 3D scene understanding. All of the mentioned methods achieve remarkable performance on 3D point cloud understanding tasks with large amounts of labeled data. In this work, we propose a self-supervised learning scheme. Almost all models mentioned above can be trained with our proposed paradigm without any human-annotated data. Then we fine-tune these models with very little labeled data on downstream tasks. Our method is label-efficient and boosts the performance of supervised models after fine-tuning on various tasks and datasets.

Unsupervised Point Cloud Understanding. Unsupervised point cloud understanding aims to capture effective information from unlabeled point cloud data and utilize the learned features to handle downstream tasks. Classic methods perform unsupervised point cloud feature learning mainly based on auto-encoders [23], [24], [25], [26], [62] and generative adversarial networks [62], [63], [64]. Despite the promising performance on several specific tasks, these methods suffer from lacking local structural supervision, which limits the feature learning ability and transferability. Certain recent efforts focus on learning both structure information and semantic knowledge by defining pretext tasks [27], [28], [30], [37], [65]. RS [27] splits the shape into 3x3x3 voxels and trains the network to reconstruct the shape whose parts have been randomly rearranged by finding the correct voxel assignment. However, RS restores the shape by simply rearranging shape parts according to the predicted voxel assignment. Such a high-level pretext task cannot constrain the model to capture effective geometry information and learn discriminative representations. Many fine-grained methods that distort and restore the shape do not apply to RS but they work well in our framework. PointGLR [28] explores high-level semantic knowledge contained in point clouds by bidirectional reasoning between local representations at different abstraction hierarchies in a network and global representation of the 3D object, which achieves extraordinary performance on classification tasks. Inspired by the great success that contrast learning achieved in 2D domain, PointContrast [31] is the first work that

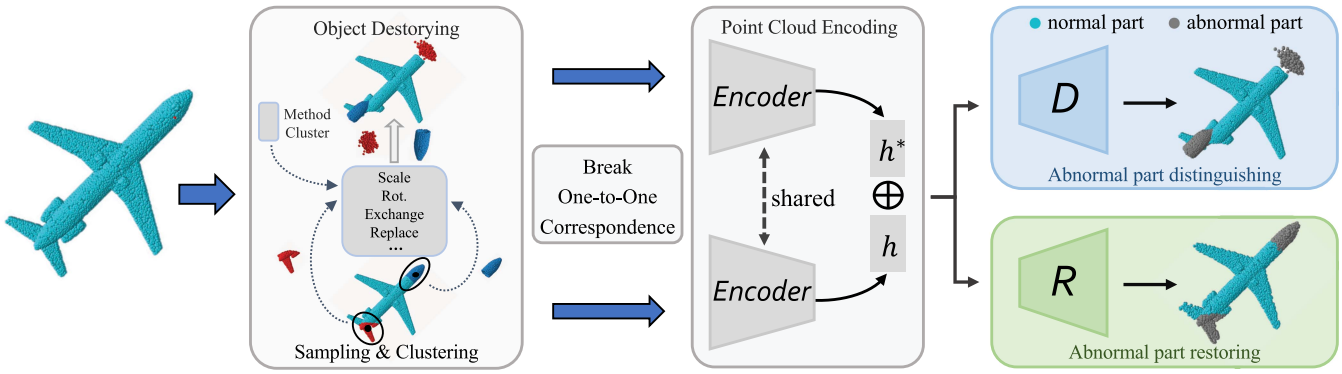

Fig. 2. Framework of the proposed self-supervised learning scheme. The framework consists of an Object-Destroying module, a point cloud encoding network and two task-related branches. We design a method cluster to distort object parts. The abnormal part distinguishing branch and abnormal part restoring branch are designed to segment points that belong to destroyed parts and restore the destroyed object to normal respectively.

proposes a contrastive self-supervised learning framework in 3D domain. It is the first time to demonstrate the transferability of learned representation in 3D point clouds to high-level scene understanding. Based on PointContrast [31], Contrastive Scene Contexts [32] integrates the spatial information into the contrastive learning framework and improves the performance over PointContrast [31], especially on complex tasks such as instance segmentation. Under this perspective, we propose a new self-supervised scheme, which simultaneously employs local and global self-supervision. Our method captures discriminative embeddings that outperform other unsupervised methods on downstream tasks and achieves better performance than state-of-the-art supervised methods after fine-tuning with all or limited labeled data.

3D Shape Assembly. Shape assembly [66], [67], [68], [69], [70], [71], [72] is another research topic that is related to our work. It aims at assembling a set of parts into a complete object. These methods usually have to predict the pose and translation of each part and learn to reconstruct the original shape. However, most of them [66], [67], [68], [69], [70], [71] use the PartNet [73] dataset, where the parts are decomposed with semantic information. In other words, additional supervisions/annotations are provided and semantic cues are taken to do shape assembly, bypassing the geometric cues. To overcome this problem, [72] proposes to cut the shape randomly following the self-supervised fashion. A Neural Shape Mating module is developed in [72] to reconstruct the shape based on the randomly decomposed parts. In general, the function of our reconstruction/restoration module is similar to the assembly module in these methods. However, the key points of shape assembly lie in predicting the pose and translation of each part, because the decomposed objects are intact and do not need to be restored. The parts in our work are broken using various methods, the model needs to find all broken parts in the shape/scene first and then restore them. Moreover, these methods only focus on shape assembly, while our method can be applied to both shape and scene data.

III. Methodology

To learn discriminative, robust and generalizable 3D representations from unlabeled point cloud data and enhance the networks ability in 3D point cloud understanding, we propose a novel self-supervised framework named learning by restoring broken geometry. Our method enables the model to capture effective structural and contextual information by destroying the local object parts and constraining the network to distinguish and restore them to normal.

A. Overview

Our framework contains an Object-Destroying module, a point cloud Encoder, a Distinguishing Branch D and a Restoring Branch R, as illustrated in Fig. 2. First, we disorganize the 3D object and destroy the geometric structure of the normal object. Then we use the encoder to generate features of both the normal object and the destroyed one. The features are concatenated as the input of branch D and R to distinguish the destroyed object parts and further restore them to normal. Here, the features of the normal object are utilized as the template to provide accurate structure information so that the network is capable of performing well on the pretext tasks, which enables the model to exploit effective object features.

Assume an object $S = \{\boldsymbol{s_1}, \boldsymbol{s_2}, \ldots, \boldsymbol{s_N}\}$ is a point set with N points, the Object-Destroying module randomly samples some parts and then utilizes a combination of approaches sampled from a destroy-method cluster to distort the sampled parts. We define the points of distorted parts as incorrect points. The incorrect points together with the parts that are not selected form a new object S^*. Intuitively, the new object may not conform the geometric characteristics of the original object. Considering the geometric characteristics implicitly represent the relationships among different object parts and imply semantic knowledge of the object, we design the Distinguishing Branch D to seek out the incorrect points that break the geometric construction of the original object, which encourages the model to better understand 3D objects and learn effective structure and semantic information. Based on the distinguishing results, if the model is able to move the incorrect points to correct positions and restore the geometric characteristics of the normal object, we can conclude that the model explores more fine-grained geometric and contextual features of input objects. Hence the Restoring Branch R is designed to reconstruct input objects. To succeed

in such a pretext task, the encoder is constrained to fully exploit useful geometry information.

B. Object Destroying

1) Destroy-Method Cluster: The Object-Destroying module is designed to destroy the geometric structure of input 3D objects by disorganizing the object parts. To simulate as many situations where objects are broken as possible, we design a method cluster to destroy the input object in this work, including (1) randomly rotate the sampled part along X,Y or Z axis; (2) randomly translate sampled points to new positions; (3) randomly scale the sampled part; (4) crop the sampled part and replace it with a random sphere; and (5) exchange the coordinates of two sampled parts.

Assume that the center point of the sampled part is $\boldsymbol{p} = (x, y, z)$. The scale e is defined as the largest distance from all points of the part to the center. For method (1), we generate a rotation matrix M given a randomly sampled rotation axis. For example, when rotating a part $P = \{\boldsymbol{p_1}, \boldsymbol{p_2}, \ldots, \boldsymbol{p_K}\}$ along Z axis, we multiple each point $\boldsymbol{p}_k$ of the part P with the rotation matrix

$$\boldsymbol{p}_{new} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{p}_k, \tag{1}$$

where $k \in \{1, 2, \ldots, K\}$ and $\theta \in [-\pi, \pi]$. For method (2), we add a random translation value t to all points of the sampled part, where the translation value $t \in [0.5e, 1.5e]$. For method (3), we first calculate the relative coordinates of each point to the center. Then the relative coordinates are multiplied with a random scale factor and added to the center to form the scaled part. The scale factor is randomly sampled from $[0.5, 1.5]$. For method (4), we generate a set of points within a sphere with the center $\boldsymbol{c} = \boldsymbol{p}$ and the radius $r \in [0.5e, 1.5e]$. Then we remove all points of the sampled part and add the points of the generated sphere. For method (5), given two sampled parts P and Q and corresponding center points $\boldsymbol{p}$ and $\boldsymbol{q}$, we first calculate relative coordinates $\boldsymbol{p}_r$ and $\boldsymbol{q}_r$ for each point. Then we exchange the two parts by adding each relative coordinate to another center:

$$\boldsymbol{p}_{new} = \boldsymbol{p}_r + \boldsymbol{q}, \tag{2}$$

$$\boldsymbol{q}_{new} = \boldsymbol{q}_r + \boldsymbol{p}. \tag{3}$$

To destroy the geometry of the sampled parts diversely, we randomly select and combine several methods for each sampled part.

These methods are based on commonly used point cloud data augmentation, and they perform very well in our work on shape datasets. For more geometrically complex and semantically rich samples such as scene point clouds, they cannot help the network to capture fine-grained information. In a large scene sample, some local parts may share similar geometric characteristics such as the local distribution of curvature while have different semantic labels. We therefore need to design some destroying methods to change its geometric characteristics (e.g., curvature) slightly and let the network find fine distinctions. When doing downstream tasks, the network will capture discriminative representations even though some parts share similar geometric characteristics.

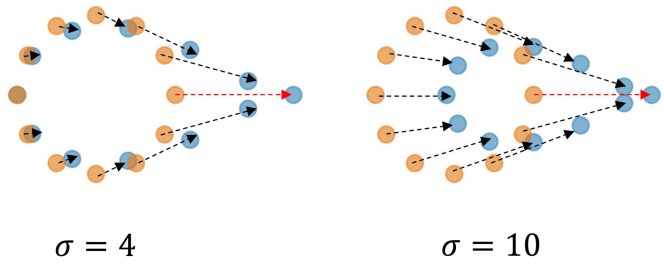

Fig. 3. Visualization of our dragging method. The red arrows denote the dragging process, while the black arrows denote the moving according to the dragging point. σ is used to normalize the scale. Best viewed in color.

To this end, we design an object-dragging method to help the model to capture more fine-grained representations and perform better on more complex tasks such as segmentation and object detection on scene datasets (as shown in Section IV-B). Just like kneading the plasticine, the object dragging is designed to drag a corner of the object to a point at 3D space. The points of the object will move along the direction of dragging, and the moving distances are inversely proportional to the distances between the points and the corner of the object. Assume that the selected corner of the object is the point $\boldsymbol{s_d} = (x_d, y_d, z_d)$. We drag the point $\boldsymbol{s_d}$ to the position $\boldsymbol{s_{d'}} = (x_{d'}, y_{d'}, z_{d'})$. All points of the object S move towards $\boldsymbol{s_{d'}}$ with distance being inversely proportional to the distance between each point and $\boldsymbol{s_d}$. As a result, the point $\boldsymbol{s_i}$ moves towards $\boldsymbol{s_{i'}} = (x_{i'}, y_{i'}, z_{i'})$, whose position is written as

$$\boldsymbol{s_{i'}} = ||\boldsymbol{s_{d'}} - \boldsymbol{s_d}|| \cdot \left(1 - \frac{||\boldsymbol{s_i} - \boldsymbol{s_d}||}{\sigma}\right) \cdot \frac{\boldsymbol{s_{d'}} - \boldsymbol{s_i}}{||\boldsymbol{s_{d'}} - \boldsymbol{s_i}||} + \boldsymbol{s_i}, \tag{4}$$

where $||\boldsymbol{s_{d'}} - \boldsymbol{s_d}|| \cdot \left(1 - \frac{||\boldsymbol{s_i} - \boldsymbol{s_d}||}{\sigma}\right)$ denotes the moving distance for point $\boldsymbol{s_i}$. σ is a hyper-parameter used to normalize the scale. $\frac{\boldsymbol{s_{d'}} - \boldsymbol{s_i}}{||\boldsymbol{s_{d'}} - \boldsymbol{s_i}||}$ in the formulation denotes the moving direction for point $\boldsymbol{s_i}$. Some geometric characteristics (e.g., curvature, size, *etc*.) of the object will be destroyed through our proposed object dragging. Moreover, we can adjust the degree of the destroying by controlling the parameter σ. The part far away from the drag point will change a little when we set the σ small. We give two 2D examples as shown in Fig. 3 to demonstrate this point. All methods described above form this destroy-method cluster. Any method used/designed to disorganize/destroy/break the local parts of an object can be added to this cluster and further enhance our self-supervised framework.

2) Random Part Sampling: For the input object, this module randomly samples some object parts and then randomly selects certain distortion approaches from the cluster to generate the destroyed object. We sample two object parts as an example to illustrate our approach. Note that our method theoretically supports any number of parts to be sampled. Specifically, we randomly select two points $\boldsymbol{s_i} = (x_i, y_i, z_i)$ and $\boldsymbol{s_j} = (x_j, y_j, z_j)$ from the input point cloud $S = \{\boldsymbol{s_1}, \boldsymbol{s_2}, \ldots, \boldsymbol{s_N}\}$ as center points.

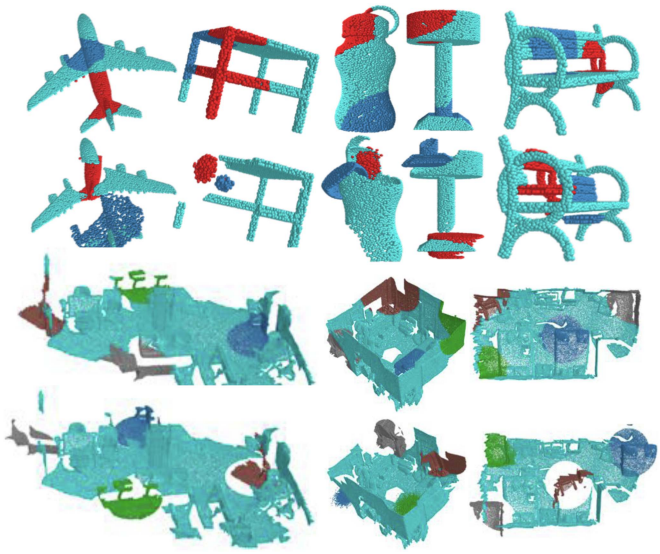

Fig. 4. Visualization of normal models (row 1 and 3) and broken models (row 2 and row 4) disorganized by Object-Destroying module.

Following the grouping layer in PointNet++, we employ ball query to sample **two** clusters of points $P = \{\boldsymbol{p_1}, \boldsymbol{p_2}, \ldots, \boldsymbol{p_K}\}$ and $Q = \{\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_K}\}$ from S, where all points in P/Q are within a radius to $\boldsymbol{s_i}$/$\boldsymbol{s_j}$ (an upper limit of K is set in our implementation). We represent the point set that is not sampled as

$$S' = S \setminus \{P \cup Q\}. \tag{5}$$

For the sampled parts P and Q, a combination of distortion approaches is utilized to generate distorted versions P^* and Q^*. Then the new object S^* can be expressed as

$$S^* = S' \cup P^* \cup Q^*, \tag{6}$$

which denotes the destroyed object that breaks the original geometric structure. The visual examples of normal objects and destroyed objects are shown in Fig. 4. Note that we explain our method with only two parts for simplicity and clarity. The number of sampled parts depends on the size of the input shape/scene. More than two parts can be sampled to perform destroying.

As shown in Fig. 2, to encourage the network to better understand the geometric characteristics of the correct object, we employ the original object as a template and the encoder extracts high-dimensional features of both the new object and original object. Intuitively, if the two objects have a point-to-point correspondence, the Distinguishing Branch tends to discern incorrect points according to transformation of coordinates rather than the geometric properties of the correct object. To avoid such correspondence in coordinates and getting trivial solutions, we use random sampling to choose two subsets of points $T = \{\boldsymbol{t_1}, \boldsymbol{t_2}, \ldots, \boldsymbol{t_{N'}}\}$, $T^* = \{\boldsymbol{t_1^*}, \boldsymbol{t_2^*}, \ldots, \boldsymbol{t_{N'}^*}\}$ from S and S^* respectively, where $N' = N/2$. Moreover, we perform simple random data augmentation on both T and T^* for the purpose of better representation learning, which further breaks the point-to-point correspondence between normal objects and disorganized objects. In the meanwhile, the Object-Destroying module generates pseudo-labels for T^*. We express it as $\mathcal{Y} = \{y_1, y_2, \ldots, y_{N'}\}$ such that $y_i \in \{0, 1\}$, where $y_i = 1$ means the corresponding point belongs to distorted parts (i.e., P^* and Q^*). The output of Object-Destroying can be expressed as a tuple $\boldsymbol{t} = [T, T^*, \mathcal{Y}]$.

C. Object Encoding

Theoretically, almost all learning-based networks that take point clouds as the input and output high-dimensional features can be utilized as the encoder of the proposed framework. In our implementation, we employ three networks, i.e., PointNet [1], RSCNN [11] and MinkowskiNet [35], as the encoder that maps input point sets from euclidean space $\mathbb{R}^{n\times 3}$ into the latent space $\mathcal{Z} \in \mathbb{R}^{n\times d}$ respectively. PointNet is a simple MLP-Based network and RSCNN is a strong point-based shape encoder that involves local and hierarchical designs. MinkowskiNet is a voxel-based network widely used in shape classification and scene segmentation. We employ three different networks to achieve superior performance and demonstrate great generality of our method. Specifically, for each object T^*, the encoder extracts its point-wise features $\boldsymbol{l^*} \in \mathbb{R}^{n\times d_l}$ and global feature $\boldsymbol{g^*} \in \mathbb{R}^{1\times d_g}$ to encode richer local and global information than the original space. When using PointNet as the encoder, global and point-wise features are defined the same as proposed in [1]. For RSCNN [11], we utilize the architecture for classification (single-scale neighborhood version) as our backbone and generate point-wise features by attaching certain feature propagation layers. For MinkowskiNet, we use the max-pooled point-wise features as the global features. For the purpose of guiding the network to correctly discern those disorganized parts, we also extract the global feature $\boldsymbol{g}$ of the original object T. The concatenation of $\boldsymbol{g}$, $\boldsymbol{g^*}$ and $\boldsymbol{l^*}$ is fed into the Distinguishing Branch D and Restoring Branch R simultaneously. Through the task of discerning the disorganized parts and restoring the original object, the encoder is encouraged to generate strong object representations that facilitate high-quality classification, segmentation, detection and other 3D point cloud understanding tasks.

D. Abnormal Part Distinguishing

For a broken object, the first task is to distinguish the destroyed parts by taking the normal object as a template/reference. Hence, distinguishing the parts that make the object violate the geometric construction enables the model to better understand 3D objects and capture more effective object features. Hence the Distinguishing Branch is designed to seek out all incorrect points of the destroyed object. We formulate the task as a point-wise classification. This task is defined as

$$\mathcal{F}_\zeta : \mathcal{Z} \in \mathbb{R}^{N'\times d} \mapsto \mathcal{Y} \in \mathbb{R}^{N'\times 2}, \tag{7}$$

which maps the high-dimensional features extracted by the point cloud encoder into predicted categories, i.e., whether the corresponding point belongs to distorted parts or not. In our method, we use PointNet/RSCNN/MinkowskiNet as the encoder, we concatenate the global features $\boldsymbol{g^*}, \boldsymbol{g} \in \mathbb{R}^{1\times d_g}$ and

the point-wise features $\boldsymbol{l}^* \in \mathbb{R}^{N' \times d_l}$ as the input of this branch. The classification is formed by several MLP layers. The output of Distinguishing Branch is denoted as $\hat{\mathcal{Y}} = \{\hat{y_1}, \hat{y_2}, \ldots, \hat{y_{N'}}\}$, where $\hat{y_i}$ represents the probability distribution formulated by softmax function.

E. Abnormal Part Restoring

The second and more challenging task is to restore the destroyed parts of the broken object. Solving this problem will help the network know more details about the 3D object and learn more discriminative representations. To this end, we develop a Restoring Branch paralleled with Distinguishing Branch and encourage the model to restore the original object. This branch aims at constraining the encoder to capture more contextual and geometric information contained in point clouds. Thus the Restoring Branch is naturally designed to move the incorrect points to original locations. We formulate the task as reconstruction based on the results of Distinguishing Branch. We define the function of Restoring Branch as

$$\mathcal{R}_\phi : \mathcal{Z} \in \mathbb{R}^{N' \times d} \mapsto \mathcal{P} \in \mathbb{R}^{N' \times 3}. \tag{8}$$

Through decoding the high-dimensional features extracted by the encoder, the Restoring Branch performs point-wise displacement prediction and tries to output a point cloud $\bar{T}$ as similar as possible to the original point set T by the function $\mathcal{R}$. Here, we use Chamfer Distance (CD) to measure the distance between the reconstructed $\bar{T}$ and the original T. The Chamfer Distance is often applied as the cost of the reconstruction task, which finds the nearest neighbor of each point and computes their euclidean distance in a bidirectional way between two point sets. In our method, considering the disorganized parts dominate the performance of reconstruction, we modify the Chamfer Distance and attach larger weights to the predicted incorrect points than the correct ones, which is written as

$$\mathcal{L}_c = \sum_{p \in T} \lambda_{\bar{p}} \min_{\bar{p} \in \bar{T}} \|p - \bar{p}\|_2^2 + \sum_{\bar{p} \in \bar{T}} \lambda_{\bar{p}} \min_{p \in T} \|p - \bar{p}\|_2^2, \tag{9}$$

where $\lambda_{\bar{p}}$ denotes the weight attached to each point in the reconstructed set. Here, we set $\lambda_{\bar{p}} \in \{0.5, 1.0\}$, where $\lambda_{\bar{p}}$ is set to 0.5 and 1.0 for points that belong to normal and distorted parts respectively. We associate the two tasks with this design, which makes them mutually reinforced. The network is also further enhanced and performs better in downstream tasks.

To accurately restore the coordinates of incorrect points, the point-wise local features $\boldsymbol{l}^*$ and global feature $\boldsymbol{g}^*$ are utilized because features of the correct points are favorable for the network to exploit the point relation information and then find proper locations of incorrect points. The same as Distinguishing Branch, we employ the global feature of the original object $\boldsymbol{g}$ as a template. Thus the input of Restoring Branch is the concatenation of $\boldsymbol{l}^*$, $\boldsymbol{g}^*$ and $\boldsymbol{g}$. The output is a reconstructed point set $\bar{T} \in \mathbb{R}^{N' \times 3}$.

F. Objective Function

The Distinguishing Branch is trained by classical cross-entropy loss and supervised by the pseudo-labels $\mathcal{Y} = \{y_1, y_2, \ldots, y_{N'}\}$, which is written as

$$\mathcal{L}_s = -\frac{1}{N'} \sum_{i=1}^{N'} y_i \log \hat{y_i}, \tag{10}$$

where $y_i \in \mathcal{Y}$ and $\hat{y_i}$ denotes the output probability distribution formulated by softmax function. We train the Restoring Branch with a modified Chamfer Distance Loss as formulated in (9).

The two branches are jointly optimized and the overall objective function of our proposed scheme is a combination of two losses

$$\mathcal{L} = \mathcal{L}_s + \beta \mathcal{L}_c, \tag{11}$$

where β is used to balance contributions of the two terms such that two branches contribute equally to the whole network.

Our common goal is to encourage the encoder to learn more discriminative object features through training it with the Restoring Broken Objects task. We define the encoder as

$$\mathcal{E}_\theta : \mathcal{P} \in \mathbb{R}^{N' \times 3} \mapsto \mathcal{Z} \in \mathbb{R}^{N' \times d}, \tag{12}$$

and any parametric non-linear function parameterized by θ can be used as the encoder. Hence the optimal problem of the proposed framework can be expressed as

$$\min_{\{\theta, \zeta, \phi\}} \mathcal{L}_s + \beta \mathcal{L}_c. \tag{13}$$

After optimization, the encoder generates more effective features and performs better on specific downstream tasks like shape classification, shape part segmentation, scene segmentation and detection.

IV. Experiments

In this section, we evaluate the proposed self-supervised learning framework qualitatively in two settings. To make a fair comparison with shape-based self-supervised methods, we train the point-based model (i.e., PointNet and RS-CNN) on the large shape dataset ShapeNet [36] with our paradigm and perform down-stream tasks (i.e., classification and part segmentation) on shape datasets. On the other hand, we also train the voxel-based model (i.e., SR-UNet) on the large-scale scene dataset ScanNet [38] and perform various down-stream tasks (i.e., segmentation and detection) to compare with recently proposed contrastive SSL frameworks [31], [32]. The summary of our experiments are presented in Fig. 5. Moreover, we conduct some ablation studies to demonstrate the effectiveness of each component introduced by our method and analyze the robustness of our method to hyper-parameters. Note that all ablation studies are performed in the first setting (i.e., shape-based.)

A. Experiments With Shape-Based Setting

Datasets. ShapeNet [36] contains more than 50,000 3D shapes across 55 categories of man-made objects. ShapeNet-Part dataset [74] contains 16,681 objects from 16 categories of ShapeNet dataset. Each category contains 2-6 parts and there are 50 parts in total. ModelNet data, i.e., ModelNet40 and ModelNet10, comprising 9832/3991 training objects and 2468/908 test objects in 40 and 10 classes respectively. ScanNet [38]

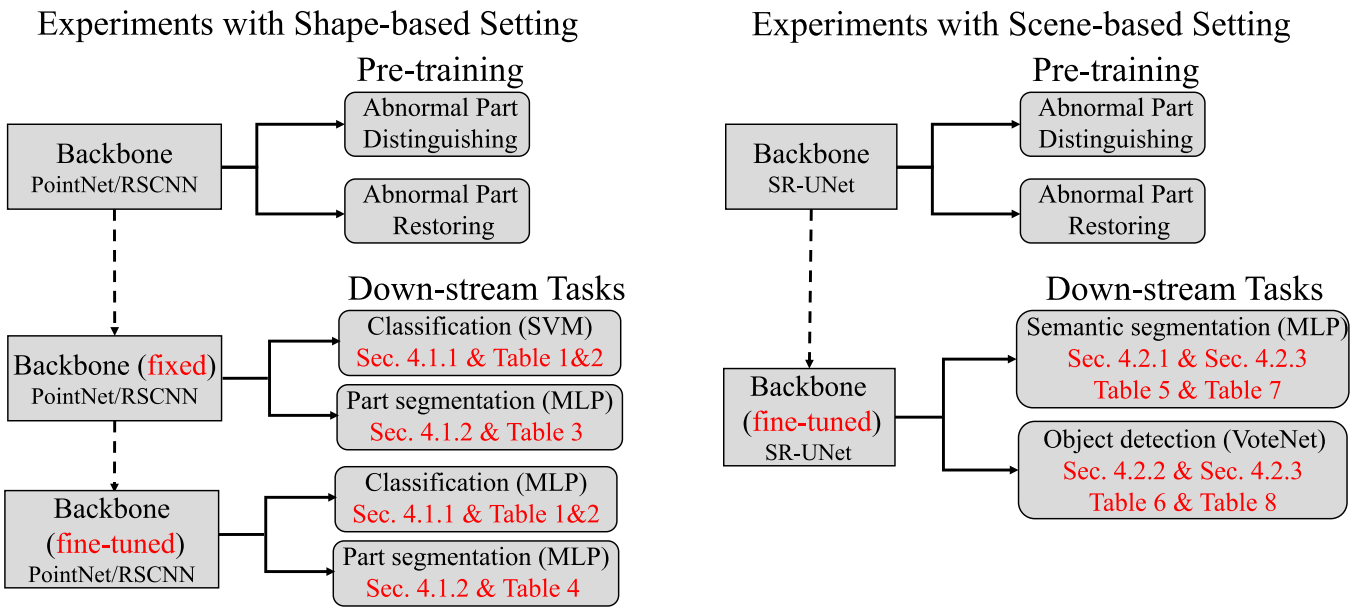

Fig. 5. The summary of experiments performed under shape-based setting and scene-based setting. Performing down-stream tasks with the fixed backbone denotes unsupervised transfer learning or training without fine-tuning. SVM denotes that we use a SVM-based classifier as the classification head. MLP denotes MLP-based classifier in classification and MLP-based point-wise classifier in segmentation. We use VoteNet as the detection architecture in scene-based experiments.

contains 1,513 scanned and reconstructed real-world indoor scenes. We follow the practice in [8], [28] to obtain shape point clouds from ScanNet according to the semantic voxel labels and form the ScanNet object dataset, which contains 17 categories.

Evaluation Metrics. For the classification task on ModelNet and ScanNet object datasets, we use classification accuracy as the metric. For segmentation on ShapeNetPart dataset, we evaluate our scheme with part classification accuracy and mean Intersection-over-Union (mIoU). For each sample, IoU is computed for each part that belongs to that object category. The mean of all part IoUs is regarded as the IoU for that sample. Ins.mIoU is defined as the mean of IoUs over all instances, and Cat. IoU is computed as an average IoU over all the instances under that category.

Model Pre-Training. Following the experimental protocol introduced in [62], we pre-train the encoder with our proposed scheme across all categories of the ShapeNet dataset, and then transfer the pre-trained model to the downstream tasks (i.e., classification on ModelNet&ScanNet object and part segmentation on ShapeNetPart). We take PointNet and RSCNN as our backbone. The Object-Destroying module, Distinguishing Branch and Restoring Branch are all discarded and only the encoder is used in downstream tasks. During pre-training, each shape in ShapeNet is sampled to 2,048 points initially. The Object-Destroying module samples two clusters of points from the input point set as stated in Section III-B and we set the upper limit number of part points K to 256. After disorganizing the input shape, we sample the new point set to 1,024 points to weaken the point-to-point correspondence between the new shape and the original one. The joint training converges within 20 epochs with Adam optimizer and batch size 32. The learning rate is set to 0.001 and the loss weight coefficient β for $\mathcal{L}_c$ is set to 4.0. Notably, only 3D coordinates are used during self-supervised training.

1) Shape Classification: To evaluate the performance of our proposed scheme on shape feature learning, we first conduct transfer experiments from ShapeNet to ModelNet/ScanNet object dataset. Following [37], [62], we extract the shape features of the ModelNet/ScanNet object samples with the pre-trained model without any parameter fine-tuning. Then we train a linear SVM on the embeddings of ModelNet/ScanNet object train split and report the classification accuracy on the ModelNet/ScanNet object test split. Each point cloud contains 1,024 points and we only use the coordinates as the input.

Results on ModelNet/ScanNet object are shown in Tables I and II ("Unsupervised Transfer Learning"). To perform fair comparisons, we reproduce PointGLR [28] without using annotated normal information as unsupervised signals. Our method achieves competitive results when only using PointNet as the encoder. When utilizing RSCNN, our method outperforms all previous unsupervised counterparts and the results on ModelNet are comparable to certain fully-supervised models. Since the pre-training of the encoder and the training of the SVM are based on different datasets, the results imply the strong transferability of our framework, which is regarded as a significant application of self-supervised representation learning. Notably, ShapeNet is a synthetic dataset sampled from CAD models and ScanNet object is a scanned real-world dataset, the domain gap between these two datasets is considered to be large. Thus the superior performance on ScanNet object further demonstrates that our model generalizes well to unseen categories and the learned features are generic.

As stated in Section II, RS [27] also disorganizes the shape and discerns the incorrect points. However, our method is motivated to offer a pipeline to destroy the geometric structure of object

TABLE I
SHAPE CLASSIFICATION RESULTS ON MODELNET. RESULTS OF BOTH SUPERVISED AND UNSUPERVISED MODELS ARE REPORTED. "UNSUPERVISED TRANSFER LEARNING" DENOTES THE PARAMETERS OF THE PRE-TRAINED MODELS ARE FIXED ON DOWNSTREAM TASKS, WHILE "SUPERVISED FINE-TUNING" DENOTES THE PRE-TRAINED MODELS ARE FINE-TUNED ON TARGET TASKS. "RI" DENOTES THE MODEL IS TRAINED ON TARGET DATASET FROM SCRATCH. "MN" DENOTES MODELNET. OUR RESULTS ARE MEASURED WITHOUT USING TRICKS LIKE VOTING

Supervision	Method	MN10 (%)	MN40 (%)
Supervised Learning	PointNet [1]	-	89.2
	PointNet++ [2]	-	90.7
	SpecGCN [75]	-	91.5
	DGCNN [3]	-	92.2
	DensePoint [10]	-	92.8
Unsupervised Transfer Learning	3D-GAN [76]	91.0	83.3
	FoldingNet [26]	94.4	88.4
	MAP-VAE [65]	94.8	90.2
	Multi-task [37]	-	89.1
	MT-PointNet [37]	-	86.2
	RS-PointNet [27]	91.6	87.3
	RS-DGCNN [27]	94.5	90.6
	GLR-RSCNN [28]	94.2	91.3
	Ours-PointNet	93.3	89.9
	Ours-RSCNN	**95.0**	**92.4**
Supervised Fine-Tuning	RI-PointNet	93.2	89.2
	Ours-PointNet	93.9**(+0.7)**	90.0(+0.8)
	RI-RSCNN	94.8	91.7
	GLR-RSCNN [28]	94.8(+0.0)	92.2(+0.5)
	Ours-RSCNN	**95.5(+0.7)**	**93.0(+1.3)**

TABLE II
SHAPE CLASSIFICATION RESULTS ON SCANNET OBJECT. THE CLASSIFICATION ACCURACY OF OUR METHOD AND THE STATE-OF-THE-ART UNSUPERVISED METHOD ARE REPORTED. "RI" DENOTES THE MODEL IS TRAINED ON SCANNET OBJECT FROM SCRATCH. WE ALSO LIST THE INCREMENTS OF PRE-TRAINING

Supervision	Method	Acc.%	Inc.%
Unsupervised Transfer	GLR-RSCNN [28]	88.1	-
	Ours-PointNet	84.2	-
	Ours-RSCNN	**89.0**	-
Supervised Fine-Tuning	RI-PointNet	87.8	-
	Ours-PointNet	89.7	+1.9
	RI-RSCNN	90.1	-
	GLR-RSCNN [28]	90.8	+0.7
	Ours-RSCNN	**92.9**	**+2.8**

parts and then distinguish and restore the distortion. We utilize a cluster of approaches to distort shape parts, which do not apply to RS. Also, we employ the features of the original shape as the template to facilitate feature learning. The Restoring Branch also contributes a lot for training the encoder, thus our method outperforms RS by a large margin.

Supervised Fine-Tuning. We think the most important application of self-supervised learning is to make full use of abundant unlabeled data and boost the performance of supervised methods. Following [31], we employ the supervised fine-tuning strategy to evaluate the effectiveness of our proposed Learning by Restoring Broken 3D Geometry. Specifically, we pre-train the model with our framework and fine-tune the weights on downstream tasks and compare the results with the randomly initialized model (not pre-trained). Under this perspective, we conduct extensive experiments on ModelNet/ScanNet object and the results are also shown in Tables I and II ("Supervised Fine-Tuning"). Note that pre-training with PointGLR [28] slightly benefits the supervised tasks while our method significantly boosts the performance, especially on ScanNet object. Pre-training with our framework can be utilized as a strong initializer for supervised models.

TABLE III
SHAPE PART SEGMENTATION RESULTS WITHOUT FINE-TUNING. PART CLASSIFICATION ACCURACY AND INS.MIOU ON SHAPENETPART DATAST ARE REPORTED. ALL COMPARED METHODS ARE EVALUATED IN A SEMI-SUPERVISED MANNER (I.E., 1%, 5% OF TRAINING DATA IS SAMPLED), WHERE THE PARAMETERS OF PRE-TRAINED MODELS ARE *FIXED*

Model	1% of train data		5% of train data	
	Accuracy	IoU	Accuracy	IoU
SO-Net	78.0	64.0	84.0	69.0
PointCapsNet	85.0	67.0	86.0	70.0
Multi-task	88.6	68.2	93.7	77.7
Ours-PointNet	84.9	69.7	88.1	74.0
Ours-RSCNN	**89.8**	**74.1**	**94.3**	**80.1**

2) Part Segmentation: Shape part segmentation is formed as a fine-grained point-wise classification task to predict the part category label of each point in a given object. Hence we explore the learned point-wise embeddings through such a task. In this section, we evaluate the learned features on ShapeNetPart dataset and report part classification accuracy and mIoU.

Following [25], [37], we first conduct the shape segmentation experiments in a semi-supervised manner, i.e., we randomly sample 1% and 5% of the ShapeNetPart train set as training data. We use the pre-trained model to extract the point features of all samples *without any parameter fine-tuning*, and then train a 4-layer MLP-based [2048, 4096, 1024, 50] classifier on the sampled training set. The evaluation is conducted on all the test samples of ShapeNetPart.

The results are shown in Table III. Our method significantly outperforms other unsupervised models [25], [37], [77], which shows that our pre-trained model captures more effective point embeddings that transfer well to segmentation tasks. Especially when using only 1% of training data, our RSCNN model outperforms all previous methods by a large margin. Considering Multi-Task [37] employs a heavier graph-based backbone, our PointNet model is also competitive. The results demonstrate that through pre-training with the proposed self-supervised scheme, a very small number of labeled samples are sufficient to achieve strong performance on the downstream task. Some results are visualized in Fig. 6. Despite the training data is limited, our model segments the fine-grained details well.

Supervised Fine-Tuning. The shape segmentation experiments under a supervised fine-tuning strategy are also conducted. We report mIoU under several training-data sampling

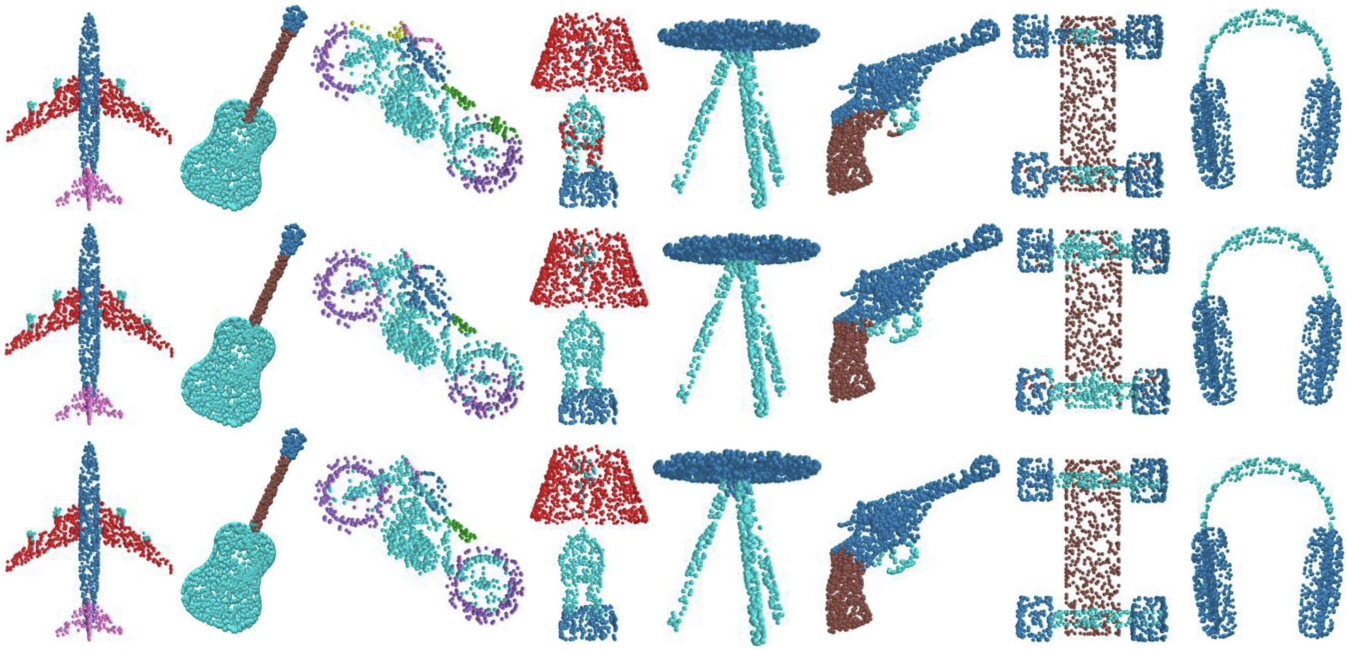

Fig. 6. The segmentation results on ShapeNetPart dataset. Row 1/2: Results predicted by the model trained on 1%/5% data with encoder *fixed*. Row 3: Ground truth.

TABLE IV
SHAPE PART SEGMENTATION RESULTS WITH FINE-TUNING STRATEGY. "RI" DENOTES THE MODEL IS NOT PRE-TRAINED. "FT" DENOTES THE MODEL IS PRE-TRAINED WITH THE CORRESPONDING UNSUPERVISED SCHEME AND FINE-TUNED ON TARGET TASK

Model	IoU (1%)	IoU (5%)	IoU (100%)
PContrast (RI)	71.8	79.3	84.7
PContrast (FT)	74.0 (+2.2)	79.9 (+0.6)	85.1 (+0.4)
Ours-PointNet (RI)	68.6	76.9	83.2
Ours-PointNet (FT)	72.9 (**+4.3**)	78.5 (**+1.6**)	84.1 (**+0.9**)
Ours-RSCNN (RI)	71.6	79.4	84.3
Ours-RSCNN (FT)	**74.3** (+2.7)	**80.4** (+1.0)	**85.2 (+0.9)**

strategies (i.e., 1%, 5%, 100%) and make comparisons with PointContrast [31] in Table IV. As shown, our RSCNN model fine-tuned on 5% labeled samples achieves a Ins.mIoU that is only 3.9% less than the fully-supervised model trained from scratch. Compared to the randomly initialized model, our pre-trained model achieves remarkable performance improvements, especially when only 1% labeled data is obtained (+4.3% for PointNet and +2.7% for RSCNN). We can conclude that pre-training with our framework on unlabeled data significantly boosts the performance and can be regarded as a strong initializer for supervised models, especially when labeled data is limited, which is a critical application of self-supervised learning.

B. Experiments With Scene-Based Setting

Datasets. ScanNet [38] is the largest indoor scene dataset, which contains a collection of 1,500 indoor scenes created with a light-weight RGB-D scanning procedure. We pre-train the model on this dataset with our proposed learning paradigm. We create a point cloud dataset on the top of ScanNet following [31]. Specifically, we sub-sample RGB-D scans from the raw ScanNet videos every 25 frames and align the 3D point clouds in the same world coordinates. A total number of 200 K point clouds are sampled to form the pre-train dataset. After pre-training, we perform various downstream tasks on Stanford Large-Scale 3D Indoor Spaces (S3DIS) [39], SUN RGB-D [40] and ScanNet [38]. S3DIS contains 6 large-scale indoor areas collected from 3 office buildings. Each point cloud is annotated with semantic labels of 13 object categories. We do the pre-processing and data augmentations following the common settings [31]. Semantic segmentation results on Area 5 and 6-fold cross-validation are reported. We also perform object detection on SUN-RGB-D, which contains about 5 K RGB-D training images annotated with amodal-oriented 3D bounding boxes for 37 object categories. The depth images are converted to point clouds using the provided camera parameters. In addition, we perform semantic segmentation and detection on ScanNet to see whether the pre-trained weights can further improve the results on ScanNet itself.

Evaluation Metrics. We use mIoU and mAcc as the metrics for scene semantic segmentation. For 3D detection, we use mAP@0.5 and mAP@0.25 as the metric.

Model Pre-Training. We pre-train our model on the ScanNet following the same setting to PointContrast [31] and ContrastivSceneContext [32] (CSC). We also use the same network for a fair comparison, i.e., a 34 layers Sparse Residual U-Net (SR-UNet) based on Minkowski convolution. In this part, we add the proposed dragging method into the destroying-method cluster. *Four* parts for each scene point cloud are sampled to break its geometric structure and the upper limit number of part

TABLE V
SEMANTIC SEGMENTATION RESULTS ON S3DIS AREA 5 (FOLD 1). PER-CATEGORY IOU, MIOU AND MACC ARE REPORTED IN THE TABLE. OUR METHOD ACHIEVES BEST MIOU AND MACC AMONG ALL SELF-SUPERVISED AND SUPERVISED METHODS. W/O DRAGGING DENOTES DESTROYING SAMPLES WITHOUT DRAGGING

Method	ceiling	floor	wall	beam	clmn	windw	door	chair	table	bkcase	sofa	board	clutter	mIOU	mAcc
PointNet [1]	88.80	97.33	69.80	0.05	3.92	46.26	10.76	52.61	58.93	40.28	5.85	26.38	33.22	41.09	48.98
SegCloud [78]	90.06	96.05	69.86	0.00	18.37	38.35	23.12	75.89	70.40	58.42	40.88	12.96	41.60	48.92	57.35
TangentConv [79]	90.47	97.66	73.99	0.0	20.66	38.98	31.34	77.49	69.43	57.27	38.54	48.78	39.79	52.8	60.7
3D RNN [80]	95.2	98.6	77.4	0.8	9.83	52.7	27.9	76.8	78.3	58.6	27.4	39.1	51.0	53.4	71.3
PointCNN [8]	92.31	98.24	79.41	0.00	17.60	22.77	62.09	80.59	74.39	66.67	31.67	62.05	56.74	57.26	63.86
MinkowskiNet20 [35]	91.55	98.49	84.99	0.8	26.47	46.18	55.82	88.99	80.52	71.74	48.29	62.98	57.72	62.60	69.62
MinkowskiNet32 [35]	91.75	98.71	86.19	0.0	34.06	48.90	62.44	89.82	81.57	74.88	47.21	74.44	58.57	65.35	71.71
PntContrast (Scratch)	91.47	98.56	84.08	0.00	33.03	56.88	63.94	90.11	81.67	72.46	76.45	77.89	59.63	68.17	75.45
PntContrast (Hardest-Ctr)	94.82	**98.72**	86.06	0.00	42.84	58.00	73.72	91.73	82.38	74.74	74.58	81.42	62.66	70.90	77.00
PntContrast (Pnt-InfoNCE)	93.26	98.67	85.56	0.11	**45.90**	54.41	67.87	91.56	80.09	74.66	78.20	81.49	62.32	70.32	76.94
ContrastiveSceneContexts	95.1	98.4	86.3	0.0	40.7	**60.8**	**85.2**	91.8	81.9	73.9	78.9	82.8	62.4	72.2	-
Ours (w/o dragging)	**95.52**	98.61	86.69	0.00	33.29	57.63	80.48	**93.78**	**82.41**	76.63	**87.39**	**86.00**	63.88	72.49	77.79
Ours	95.30	98.57	**87.42**	0.00	37.53	58.50	81.46	92.50	82.11	**77.88**	85.53	85.55	**65.16**	**72.89**	**78.93**

points K is set to 4,096. We use SGD with learning rate and batch size to be 0.1 and 32 respectively to optimize the model for 60 K steps. The learning rate is decreased by a factor of 0.99 every 1,000 steps. Then we fine-tune the pre-trained model on various downstream tasks to prove effectiveness of our proposed self-supervised method.

1) Semantic Segmentation on S3DIS: We use the widely taken Area 5 Test split for training and testing. We train the model using the SGD with learning rate to be 0.8 and batch size to be 48. The Polynomial LR scheduler with a power factor of 0.9 is taken to optimize the network. We report the results on Table V, from which we see that our method achieves an improvement of *4.72* mIoU compared with training from scratch. In the meantime, our method even performs better than PointContrast and CSC, which are state-of-the-art self-supervised methods for scene point clouds. Note that PointContrast and CSC need to elaborately construct positive-positive and positive-negative pairs and design metric object functions for training. However, our method is easy to be implemented and robust to various hyper-parameters (as shown in Section IV-C). Moreover, our method achieves state-of-the-art results compared with other supervised point cloud segmentation methods and performs best on most categories among 13 classes. These results fully demonstrate that our proposed self-supervised pre-training method does capture rich geometric and semantic information from 3D scenes and has great generalization ability. We also report the results that we do not add the dragging method into the destroying cluster and prove that more fine-grained destroying method does improve the tasks on scene datasets. We present some visualization results in Fig. 7 to further prove that our method does achieve significantly better performance than the model trained from random initialization.

2) Object Detection on SUN RGB-D: Object detection is a more complex task compared with semantic segmentation, which needs the network to predict bounding boxes and their corresponding category labels. Following the setting used in PointContrast and CSC, we take the strongest detection framework VoteNet [81] and replace its backbone network with SR-UNet. We perform this kind of downstream task first on SUN RGB-D dataset and report the results in Table VI. Our method achieves better mAP@0.5 than state-of-the-art self-supervised method CSC and ranks only second to CSC using mAP@0.25 as metric. Results without dragging are also shown in Table VI. We show some visualization results in Fig. 8.

TABLE VI
OBJECT DETECTION RESULTS ON SUN RGB-D. OUR METHOD ACHIEVES THE BEST MAP@0.5 PERFORMANCE AND RANKS SECOND USING THE MAP@0.25 AS METRIC

Method	mAP@0.5	mAP@0.25
VoteNet [81]	32.9	57.7
Train from scratch	31.7	55.6
PointContrast (Hardest-Contrastive)	34.5	57.5
PointContrast (PointInfoNCE)	34.8	57.5
ContrastiveSceneContexts	36.4	**58.9**
Ours (w/o dragging)	36.7	57.6
Ours	**37.3**	58.2

TABLE VII
SEMANTIC SEGMENTATION RESULTS ON SCANNET VALIDATION SET. OUR METHOD ACHIEVES COMPARABLE RESULTS TO CSC

Method	mIoU	mAcc
Train from scratch	72.2	80.7
PointContrast (Hardest-Contrastive)	73.3	81.0
PointContrast (PointInfoNCE)	**74.1**	**81.6**
ContrastiveSceneContexts	73.8	81.3
Ours (w/o dragging)	73.4	80.8
Ours	73.8	81.1

3) Semantic Segmentation and Object Detection on ScanNet: We perform semantic segmentation and object detection on ScanNet v2, which is generated from ScanNet dataset used for pre-training. Our method further improves the downstream tasks on ScanNet even pre-training the model on the dataset from the same source. For semantic segmentation, we also use the SR-UNet to perform per-point prediction. For object detection, we follow VoteNet [81] and replace the backbone network with our trained SR-UNet. Experimental results are summarized in Tables VII and VIII, from which we see that our method achieves better or comparable results to state-of-the-art self-supervised methods. Especially for object detection, our method improves mAP@0.5 by *1.4* and mAP@0.25 by *1.0* compared with the strongest method ever CSC. We also provide

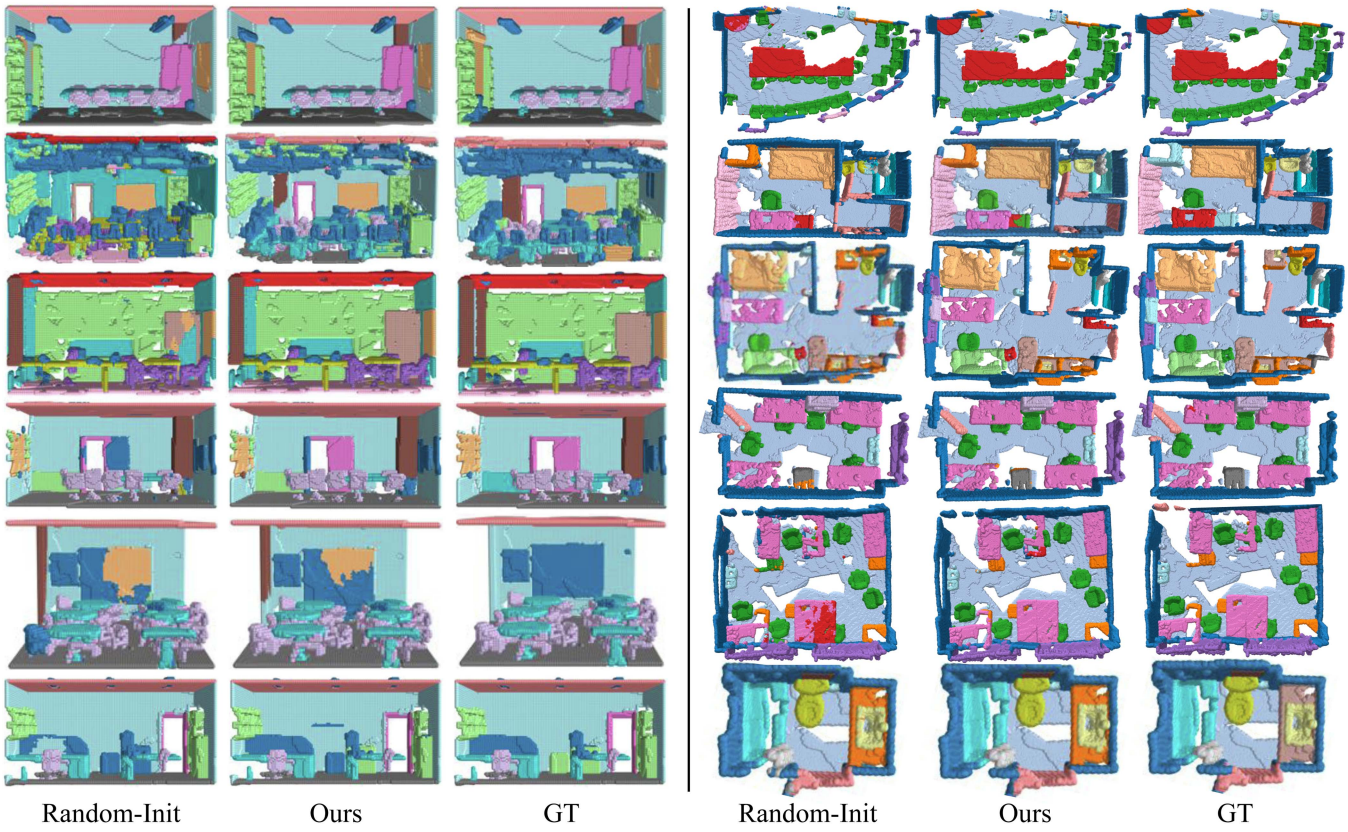

Fig. 7. Visualization results for semantic segmentation on S3DIS and ScanNet. We present results of S3DIS and ScanNet on column 1-3 and column 4-6 respectively. Our method improves the model trained from random initialization significantly.

TABLE VIII
OBJECT DETECTION RESULTS ON SCANNET VALIDATION SET. OUR METHOD ACHIEVES THE BEST RESULTS COMPARED WITH ALL SELF-SUPERVISED METHODS PRESENTED IN THE TABLE

Method	mAP@0.5	mAP@0.25
DSS [82]	6.8	15.2
3D-SIS [83]	22.5	40.2
VoteNet [81]	33.5	58.6
Train from scratch	35.4	56.7
PointContrast (Hardest-Contrastive)	37.3	59.2
PointContrast (PointInfoNCE)	38.0	58.5
ContrastiveSceneContexts	39.3	59.1
Ours (w/o dragging)	39.5	59.6
Ours	**40.7**	**60.1**

TABLE IX
RELATIVE PERFORMANCE GAINS OF POINTNET AND RSCNN. ALL RESULTS IN THIS TABLE ARE CALCULATED UNDER THE FINE-TUNED SETTING. "R-INC" DENOTES RELATIVE PERFORMANCE IMPROVEMENTS

Dataset	Task	R-Inc.% (PN)	R-Inc.% (RSCNN)
ModeNet 10	Classification	0.75	0.73
ModeNet 40	Classification	0.89	1.40
ScanNet Object	Classification	2.12	3.01
ShapeNetPart	Segmentation	1.07	1.06

some visualization results of segmentation and detection in Figs. 7 and 8 respectively.

C. Ablation Study

In this section, we explore the crucial components and hyper-parameters of our self-supervised learning framework. All the experiments in this section are conducted on ModelNet40 dataset and we fix the encoder (RSCNN) after pre-training.

Component Analyses. We first analyze the relationship between the improvements of our method on downstream tasks and the capacity/style of different models. Then we conduct ablation study to investigate the effectiveness of each branch in our framework. We remove the corresponding loss when investigating the effect of such a branch. Besides, we perform points down-sampling and data augmentation to break the coordinate correspondence. Hence we also conduct experiments to explore the effectiveness of such operations.

RSCNN models local geometry information and has a hierarchical structure, giving it a larger capacity and better representation ability than PointNet. As shown in Table IX, we report the relative performance improvements of PointNet and RSCNN in different datasets and tasks. We observe that RSCNN transfers a little better to most down-stream tasks than PointNet. The reasons may lie in that larger models are easier to overfit on down-stream tasks and our proposed self-supervised learning method alleviates this problem. Even for a large model, it is hard

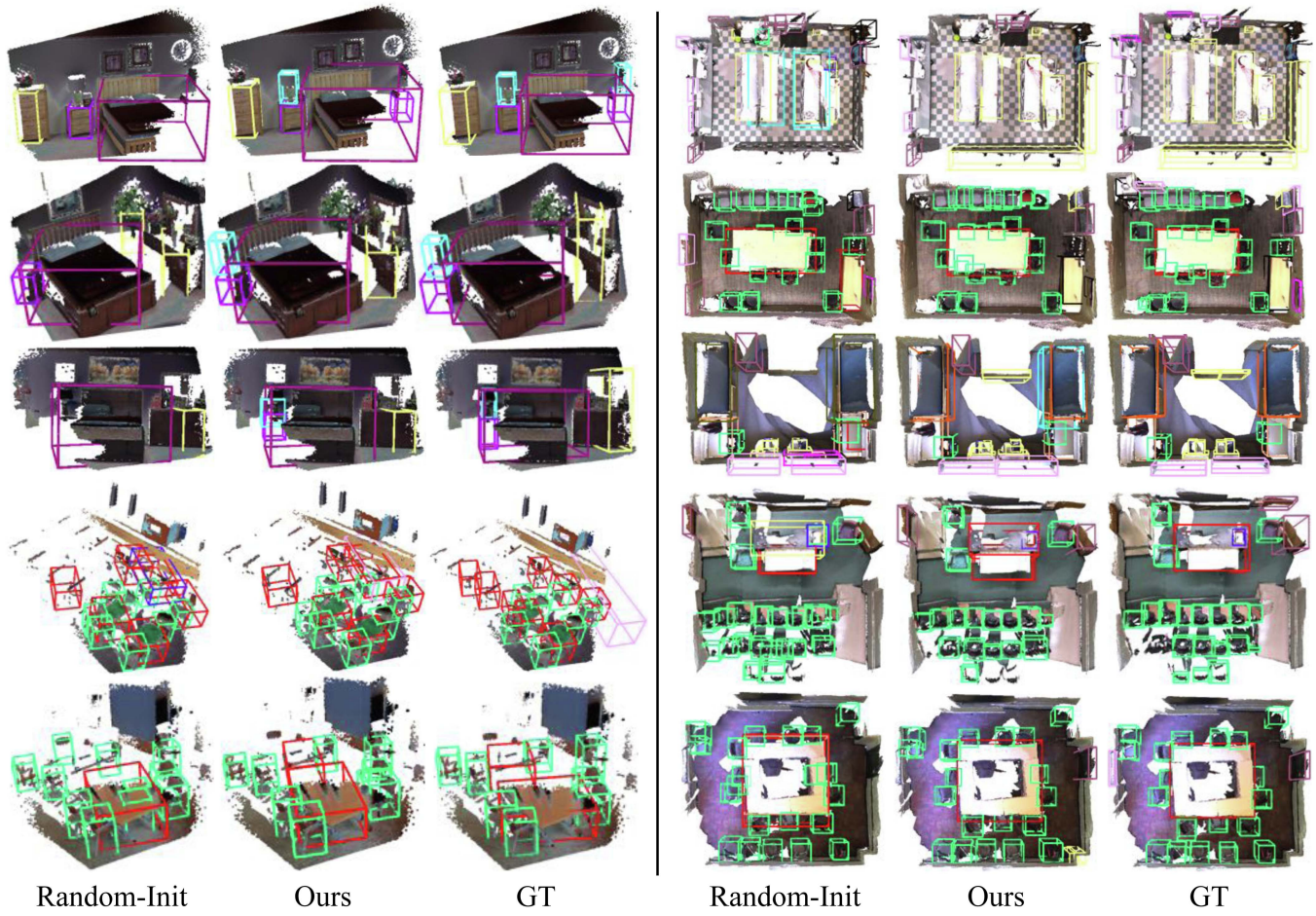

Fig. 8. Visualization results for object detection on SUN RGB-D and ScanNet. We present results of SUN RGB-D and ScanNet on column 1-3 and column 4-6 respectively. We observe that the model pre-trained with our method achieves more accurate object detection than the model trained from random initialization.

to overfit our designed pre-text tasks (destroying and restoring), especially performing on a large scale and unlabeled dataset. Thereby, pre-training large models on our proposed pre-text tasks endow the models with strong and generalizable representation ability. Initializing the down-stream models with the pre-trained parameters makes it harder to overfit down-stream tasks. In summary, large models are easier to be overfitted than small models, while our proposed self-supervised paradigm alleviates this overfitting problem and brings more improvements for large models when compared with small models.

The results shown in Table X indicate that the Distinguishing Branch plays a more important role than the Restoring Branch, while Restoring Branch can further improve performance. We also compare the ablated version without features from the template shape and the accuracy degrades to 87.8%, which convincingly verifies the effectiveness of utilizing the features of original shapes. Moreover, we verify the effectiveness of our modified CD loss as described in Section III-E by replacing it with the original version. The results show that our modified CD loss improves the original version by 0.5% accuracy for classification on ModelNet40.

Finally, we explore how the approach cluster in Object-Destroying module affects the performance of the scheme. The

TABLE X
COMPONENT ANALYSES. ACCURACY RESULTS ON MODELNET40 ARE SHOWN

Branch D	Branch R	Aug.& Down-sample	Temp Feature	Modified CD loss	Acc. %
	√	√	√	√	88.1
√		√	√	√	89.3
√	√		√	√	90.9
√			√	√	88.0
	√		√	√	87.6
√	√	√		√	87.8
√	√	√	√		91.9
√	√	√	√	√	**92.4**

results are shown in Table XI. Notably, our method achieves competitive performance by only randomly translating and rotating sampled parts, which strongly demonstrates the effectiveness of our pipeline. We also generate abnormal objects by only adding noise to the original shapes and the accuracy degrades to 87.2%, which proves the importance of altering geometric structure on the pre-task.

Parameter Analyses. In our implementation, we randomly select 2 center points for the shape setting. Then we sample K points within a radius to each center. In fact, if we further

TABLE XI
EFFECTIVENESS OF THE DISTORTION APPROACHES. ACCURACY RESULTS ON MODELNET40 ARE SHOWN

Rot.	Trans.	Scale.	Exchange.	Replace.	Acc.%
√	√				89.2
√	√	√			89.5
			√		91.1
				√	90.7
			√	√	92.0
√	√	√	√	√	**92.4**

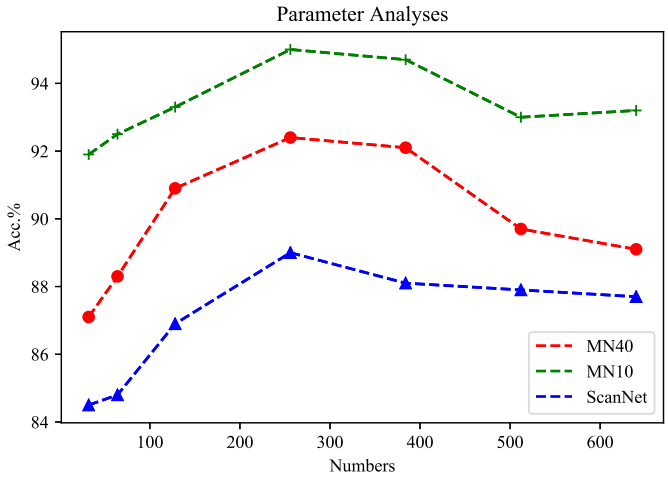

Fig. 9. Parameter analyses on the point numbers of each distorted part. When the number of destroyed parts is set to 2, we achieve best performance by setting the point numbers of each distorted part to 256.

increase the number of selected center points, K needs to be elaborately designed. Smaller K fails to break the geometric construction of each selected local region, while larger K (e.g., 4 parts and K = 256 select 1024 points) completely destroys the global structure of the shape, which also has a negative effect on our task. Hence, 2 is an appropriate number of the selected parts. We explore how the number of incorrect points (i.e., the hyper-parameter K as stated in Section III-B) affects the performance of the model in this part. The results are shown in Fig. 9. We can observe that good performance is achieved when K is set to 256. No obvious improvements show up when further increasing K.

V. CONCLUSION

We propose a self-supervised framework for point cloud analysis named Learning by Restoring Broken 3D Geometry. Large amounts of experiments are performed with shape setting and scene setting respectively on various datasets. This method is based on an intuitive principle that a network is strong enough to extract discriminative representation of 3D model when it has ability to restore broken 3D models. Then a framework containing abnormal part distinguishing and restoring module is designed to perform self-supervised learning. Experiments with both shape setting and scene setting are performed to demonstrate effectiveness of our proposed framework. Experimental results prove that our method transfers well to various downstream tasks and achieves state-of-the-art performance among almost all un-/self-supervised methods. Notably, Learning by Restoring Broken 3D Geometry can be regarded as a general pipeline and we provide a simple and effective implementation for point clouds. It is an easily extensible framework and has great compatibility.

REFERENCES

[1] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.
[2] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.
[3] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.
[4] H. Su et al., "SPLATNet: Sparse lattice networks for point cloud processing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2530–2539.
[5] S. Xie, S. Liu, Z. Chen, and Z. Tu, "Attentional shapecontextnet for point cloud recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4606–4615.
[6] Y. Shen, C. Feng, Y. Yang, and D. Tian, "Mining point cloud local structures by kernel correlation and graph pooling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4548–4557.
[7] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 984–993.
[8] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on x-transformed points," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 820–830.
[9] S. Lan, R. Yu, G. Yu, and L. S. Davis, "Modeling local geometric structure of 3D point clouds using geo-CNN," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 998–1008.
[10] Y. Liu, B. Fan, G. Meng, J. Lu, S. Xiang, and C. Pan, "DensePoint: Learning densely contextual representation for efficient point cloud processing," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 5239–5248.
[11] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8895–8904.
[12] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "ShellNet: Efficient point cloud convolutional neural networks using concentric shells statistics," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1607–1616.
[13] Q. Huang, W. Wang, and U. Neumann, "Recurrent slice networks for 3D segmentation of point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2626–2635.
[14] J. Liu, B. Ni, C. Li, J. Yang, and Q. Tian, "Dynamic points agglomeration for hierarchical point sets learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7546–7555.
[15] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2536–2544.
[16] R. Zhang, P. Isola, and A. A. Efros, "Split-brain autoencoders: Unsupervised learning by cross-channel prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 645–654.
[17] H. Mobahi, R. Collobert, and J. Weston, "Deep learning from temporal coherence in video," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 737–744.
[18] D. Jayaraman and K. Grauman, "Slow and steady feature analysis: Higher order temporal coherence in video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3852–3861.
[19] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5729–5738.
[20] K. He, H. Fan, Y. Wu, S. Xie, and R. B. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9726–9735.
[21] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[22] J. Grill et al., "Bootstrap your own latent: A new approach to self-supervised learning," 2020, *arXiv:2006.07733*.

[23] H. Deng, T. Birdal, and S. Ilic, "PPF-FoldNet: Unsupervised learning of rotation invariant 3D local descriptors," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 602–618.

[24] Y. Sun, Y. Wang, Z. Liu, J. Siegel, and S. Sarma, "PointGrow: Autoregressively learned point cloud generation with self-attention," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2020, pp. 61–70.

[25] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, "3D point capsule networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1009–1018.

[26] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 206–215.

[27] J. Sauder and B. Sievers, "Self-supervised deep learning on point clouds by reconstructing space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 12962–12972.

[28] Y. Rao, J. Lu, and J. Zhou, "Global-local bidirectional reasoning for unsupervised representation learning of 3D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5376–5385.

[29] M. Gadelha et al., "Label-efficient learning on point clouds using approximate convex decompositions," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 473–491.

[30] I. Achituve, H. Maron, and G. Chechik, "Self-supervised learning for domain adaptation on point clouds," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2021, pp. 123–133.

[31] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany, "PointContrast: Unsupervised pre-training for 3D point cloud understanding," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 574–591.

[32] J. Hou, B. Graham, M. Nießner, and S. Xie, "Exploring data-efficient 3D scene understanding with contrastive scene contexts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15582–15592.

[33] Z. Zhang, R. Girdhar, A. Joulin, and I. Misra, "Self-supervised pretraining of 3D features on any point-cloud," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10232–10243.

[34] P. Wang, Y. Yang, Q. Zou, Z. Wu, Y. Liu, and X. Tong, "Unsupervised 3D learning for shape analysis via multiresolution instance discrimination," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 2773–2781.

[35] C. B. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convnets: Minkowski convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3070–3079.

[36] A. X. Chang et al., "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*.

[37] K. Hassani and M. Haley, "Unsupervised multi-task feature learning on point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 8160–8171.

[38] A. Dai, A. X. Chang, M. Savva, M. Halber, T. A. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2432–2443.

[39] I. Armeni et al., "3D semantic parsing of large-scale indoor spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1534–1543.

[40] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 567–576.

[41] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 1422–1430.

[42] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 69–84.

[43] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 649–666.

[44] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," 2018, *arXiv:1803.07728*.

[45] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *Proc. Eur. Conf. Comput. Vis.*, A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, Eds. Springer, 2020, pp. 776–794.

[46] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 2794–2802.

[47] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: Unsupervised learning using temporal order verification," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 527–544.

[48] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 87–102.

[49] B. Graham, M. Engelcke, and L. Van Der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9224–9232.

[50] P. Hermosilla, T. Ritschel, P.-P. Vázquez, À. Vinacua, and T. Ropinski, "Monte Carlo convolution for learning on non-uniformly sampled point clouds," *ACM Trans. Graph.*, vol. 37, no. 6, pp. 1–12, 2018.

[51] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, and R. Urtasun, "Deep parametric continuous convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2589–2597.

[52] L. Yi, H. Su, X. Guo, and L. J. Guibas, "SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2282–2290.

[53] G. Te, W. Hu, A. Zheng, and Z. Guo, "RGCNN: Regularized graph CNN for point cloud segmentation," in *Proc. 26th ACM Int. Conf. Multimedia*, 2018, pp. 746–754.

[54] D. Rethage, J. Wald, J. Sturm, N. Navab, and F. Tombari, "Fully-convolutional point networks for large-scale point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 596–611.

[55] W. Wang, R. Yu, Q. Huang, and U. Neumann, "SGPN: Similarity group proposal network for 3D point cloud instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2569–2578.

[56] Y. Wang, D. J. Tan, N. Navab, and F. Tombari, "SoftPoolNet: Shape descriptor for point cloud completion and classification," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 70–85.

[57] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9621–9630.

[58] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6411–6420.

[59] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel CNN for efficient 3D deep learning," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, Art. no. 87.

[60] H. Tang et al., "Searching efficient 3D architectures with sparse point-voxel convolution," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 685–702.

[61] Y. Lin, Z. Zhang, H. Tang, H. Wang, and S. Han, "PointAcc: Efficient point cloud accelerator," in *Proc. IEEE/ACM 54th Annu. Int. Symp. Microarchitecture*, 2021, pp. 449–461.

[62] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2018, pp. 40–49.

[63] C. Li, M. Zaheer, Y. Zhang, B. Póczos, and R. Salakhutdinov, "Point cloud GAN," in *Proc. Deep Generative Models Highly Structured Data Workshop*, New Orleans, LA, USA, 2019. [Online]. Available: https://openreview.net/forum?id=S1xim8UFuV

[64] D. Valsesia, G. Fracastoro, and E. Magli, "Learning localized generative models for 3D point clouds via graph convolution," in *Proc. 7th Int. Conf. Learn. Representations*, New Orleans, LA, USA, 2019. [Online]. Available: https://openreview.net/forum?id=SJeXSo09FQ

[65] Z. Han, X. Wang, Y.-S. Liu, and M. Zwicker, "Multi-angle point cloud-VAE: Unsupervised feature learning for 3D point clouds from multiple angles by joint self-ction and half-to-half prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 10441–10450.

[66] A. Narayan, R. Nagar, and S. Raman, "RGL-NET: A recurrent graph learning framework for progressive part assembly," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2022, pp. 78–87.

[67] G. Zhan et al., "Generative 3D part assembly via dynamic graph learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 6315–6326.

[68] B. Jones, D. Hildreth, D. Chen, I. Baran, V. G. Kim, and A. Schulz, "AutoMate: A dataset and learning approach for automatic mating of CAD assemblies," *ACM Trans. Graph.*, vol. 40, no. 6, pp. 1–18, 2021.

[69] Y. Li, K. Mo, L. Shao, M. Sung, and L. Guibas, "Learning 3D part assembly from a single image," in *Proc. 16th Eur. Conf. Comput. Vis.*, Glasgow, U.K., Springer, 2020, pp. 664–682.

[70] K. D. Willis et al., "JoinABLe: Learning bottom-up assembly of parametric cad joints," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 15849–15860.

[71] K. Yin, Z. Chen, S. Chaudhuri, M. Fisher, V. G. Kim, and H. Zhang, "COALESCE: Component assembly by learning to synthesize connections," in *Proc. IEEE Int. Conf. 3D Vis.*, 2020, pp. 61–70.

[72] Y.-C. Chen, H. Li, D. Turpin, A. Jacobson, and A. Garg, "Neural shape mating: Self-supervised object assembly with adversarial shape priors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12724–12733.

[73] K. Mo et al., "PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 909–918.
[74] L. Yi et al., "A scalable active framework for region annotation in 3D shape collections," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–12, 2016.
[75] C. Wang, B. Samari, and K. Siddiqi, "Local spectral graph convolution for point set feature learning," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 52–66.
[76] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 82–90.
[77] J. Li, B. M. Chen, and G. Hee Lee, "SO-net: Self-organizing network for point cloud analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9397–9406.
[78] L. P. Tchapmi, C. B. Choy, I. Armeni, J. Gwak, and S. Savarese, "SEGCloud: Semantic segmentation of 3D point clouds," in *Proc. Int. Conf. 3D Vis.*, 2017, pp. 537–547.
[79] M. Tatarchenko, J. Park, V. Koltun, and Q. Zhou, "Tangent convolutions for dense prediction in 3D," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3887–3896.
[80] X. Ye, J. Li, H. Huang, L. Du, and X. Zhang, "3D recurrent neural networks with context fusion for point cloud semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., 2018, pp. 415–430.
[81] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3D object detection in point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9276–9285.
[82] S. Song and J. Xiao, "Deep sliding shapes for amodal 3D object detection in RGB-D images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 808–816.
[83] J. Hou, A. Dai, and M. Nießner, "3D-SIS: 3D semantic instance segmentation of RGB-D scans," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4416–4425.

Jinxian Liu received the BEng degree in electronic engineering from Tianjin University. He is currently working toward the PhD degree with Shanghai Jiao Tong University under the supervision of *Prof.* Bingbing Ni. His research interests mainly lie in 3D computer vision, graphics and physics-based simulation.

Bingbing Ni received the BEng degree in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2005, and the PhD degree from the National University of Singapore, Singapore, in 2011. He is currently a professor with the Department of Electrical Engineering, Shanghai Jiao Tong University. Before that, he was a research scientist with the Advanced Digital Sciences Center, Singapore. He was with Microsoft Research Asia, Beijing, China, as a research intern, in 2009. He was also a software engineer intern with Google Inc., Mountain View, California , in 2010. He was a recipient of the Best Paper Award from PCM'11 and the Best Student Paper Award from PREMIA'08. He was also the recipient of the first prize in the International Contest on Human Activity Recognition and Localization in conjunction with the International Conference on Pattern Recognition, in 2012.

Ye Chen received the BEng degree in electronic engineering from Shanghai Jiao Tong University. He is currently working toward the master's degree with Shanghai Jiao Tong University under the supervision of *Prof.* Bingbing Ni. His research interests mainly lie in 3D vision and physics-based simulation.

Zhenbo Yu received the MS degree from the Nanjing University of Information Science and Technology, Nanjing, China, in 2016. He is currently working toward the PhD degree with Shanghai Jiao Tong University. His current research interests include 3D pose estimation and 3D human mesh recovery.

Hang Wang received the BEng and master's degrees in electronic engineering from Shanghai Jiao Tong University under the supervision of *Prof.* Bingbing Ni. He is currently an algorithm researcher with Huawei Hisilicon, Shanghai, China. His research interests mainly lie in visual recognition and transfer learning.