Learning Fine-Grained Segmentation of 3D Shapes without Part Labels

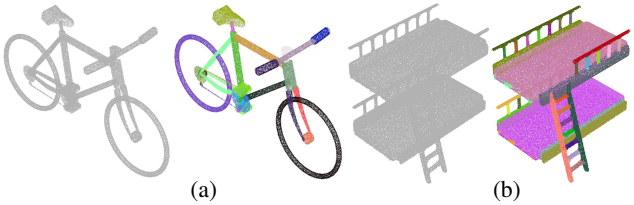
Xiaogang Wang[1,2] Xun Sun[2] Xinyu Cao[2] Kai Xu[3*] Bin Zhou[2*]
[1]Southwest University
[2]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University
[3]National University of Defense Technology

Abstract

Learning-based 3D shape segmentation is usually formulated as a semantic labeling problem, assuming that all parts of training shapes are annotated with a given set of tags. This assumption, however, is impractical for learning fine-grained segmentation. Although most off-the-shelf CAD models are, by construction, composed of fine-grained parts, they usually miss semantic tags and labeling those fine-grained parts is extremely tedious. We approach the problem with deep clustering, where the key idea is to learn part priors from a shape dataset with fine-grained segmentation but no part labels. Given point sampled 3D shapes, we model the clustering priors of points with a similarity matrix and achieve part segmentation through minimizing a novel low rank loss. To handle highly densely sampled point sets, we adopt a divide-and-conquer strategy. We partition the large point set into a number of blocks. Each block is segmented using a deep-clustering-based part prior network trained in a category-agnostic manner. We then train a graph convolution network to merge the segments of all blocks to form the final segmentation result. Our method is evaluated with a challenging benchmark of fine-grained segmentation, showing state-of-the-art performance.

1. Introduction

3D shape segmentation is a fundamental problem in 3D vision. While most existing works focus on semantic segmentation of 3D shapes into major parts (e.g., seat, back and leg of a chair), many application scenarios, on the other hand, demand fine-grained shape segmentation. A definition of fine-grained parts was given in [22]. In that work, fine-grained parts are defined in contrast with semantic parts. While semantic parts are major, functional ones (e.g., the back, seat and leg parts of a chair), fine-grained parts mainly refer to modeling components which are, albeit geometrically insignificant, conceptually meaningful in the sense of assembly-based 3D modeling. Therefore, fine-grained segmentation induces an intricate structural analysis of 3D objects, which facilitates part-based shape synthesis and modeling [13, 25] and meticulous robotic manipulation [1, 6]. Consequently, the problem receives increasing research attention lately [28, 23], along with dedicated datasets [14].



(a) (b)

Figure 1: Two examples of fine-grained segmentation. For each example, the left is the input point cloud and the right is the fine-grained segmentation result.

Previous learning-based approaches to fine-grained 3D shape segmentation usually formulate it as a semantic labeling problem. This requires a large training dataset of 3D shapes with fine-grained part segmentation and tags. When working with most online shape repositories such as ShapeNet [3], fine-grained part segmentation comes for free since most off-the-shelf CAD models are, by construction, composed of fine-grained parts. These fine-grained parts, however, have no, or noisy semantic tags. Annotating fine-grained parts with semantic tags is extremely tedious due to the tiny part size and large part count (range from tens to hundreds; see [23] for statistics). Moreover, many fine-grained parts may not even have a well-defined tag. Due to these reasons, the existing fine-grained part datasets [14] does leave many parts unlabeled.

In this paper, we introduce a *deep clustering* based approach to fine-grained part segmentation, thus avoiding the requirement of part labels. The key idea is to learn *geometric part priors* describing *what constitutes a fine-grained part*, based on a shape dataset with fine-grained segmentation but no part labels. Working with point sampled 3D shapes, our method models the clustering priors of 3D points with a similarity matrix of point features capturing

*Corresponding author: kevin.kai.xu@gmail.com,zhoubin@buaa.edu.cn

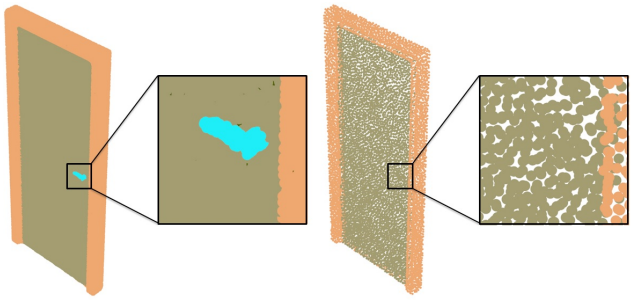


Figure 2: Tiny parts demand high sampling rate for accurate fine-grained segmentation.

for any two points how likely they belong to the same part. This similarity matrix possesses low rank property with the rank equal to the number of fine-grained parts of the shape. Therefore, fine-grained part segmentation can be achieved by minimizing a novel low rank loss over the similarity matrix.

Fine-grained parts are usually very tiny compared to the full shape (see Figure 2 (a)). A moderate sampling rate of 3D shapes can hardly capture the geometry of tiny parts accurately, which can result in suboptimal segmentation (see Figure 2 (b)). Therefore, fine-grained segmentation needs to work with densely sampled shapes. Existing deep learning models for 3D point clouds, such as PointNet [15], usually find difficulty in handling very large point clouds. To this end, we adopt a divide-and-conquer strategy. We first partition the large point set into a number of blocks. Each block is segmented using a deep-clustering-based part prior network, called PriorNet, which is trained in a category-agnostic manner. Benefiting from the block-wise training strategy, the required training shapes are greatly reduced. We then train MergeNet, a graph convolution network, to merge the segments of all blocks to form the final segmentation.

The main contributions of our paper include:

- a deep-clustering-based formulation for fine-grained segmentation of 3D shapes which learns geometric part priors without relying on part annotations,
- a novel low-rank loss designed for learning fine-grained part priors, and
- a novel graph convolution network based module trained to merge segments in different blocks.

2. Related Work

Point cloud segmentation. Point cloud segmentation has gained significant research progress in recent year, benefiting from the advances in machine learning techniques [19, 16, 9, 21]. Early studies [17, 18, 10, 5] most utilize hand-crafted features towards specific tasks. These features often encode statistical properties of points and are designed to be invariant transformations, which can be categorized as local features [2, 20] and global features [18, 10, 7, 5]. For a specific task, it is not trivial to find the optimal feature combination.

Recently many deep learning architectures have been developed for point cloud data [19, 16, 9]. These methods demonstrate remarkable performance in part segmentation of object and scene segmentation. All these models, however, find difficult in handling large point cloud. In general, the point sets is down-sampled at first, which is used as the input as the input of the neural network. However, after down-sampling, many fine-grained parts are lost. To our knowledge, very few works have studied fine-grained segmentation of point clouds. Mo et al. [14] collected a large-scale dataset with manually annotated fine-grained semantic parts. They also proposed some baseline methods for fine-grained segmentation of 3D point cloud. Luo et al. [12] introduced a data-driven iterative perceptual grouping pipeline for the task of zero-shot 3D shape part discovery. Yu et al. [28] proposed a top-down recursive decomposition network for fine-grained segmentation of 3D point cloud. However, their methods require well-defined fine-grained part semantic lables.

Low rank representation and loss. Low rank representation is a robust and efficient tool for processing high-dimensional data. This is because low rank representation has an excellent performance in discovering global structures of data. The low-rank representation can reveal the relationships of the samples: the within-cluster affinities are dense while the between-cluster affinities are all zeros [11]. Low rank representation has been widely used in many applications of image processing including image denoising [24], face recognition [4], and classification [29] in recent years.

Recently, Yi et al. [27] introduced a low rank loss based on deep learning for estimating detailed scene illumination using human faces in a single image. The strategy based on the observation that the diffuse chromaticity over a face should be consistent among images, regardless of illumination changes, because a person's facial surface features should remain the same. The diffuse chromaticity of multiple aligned images of the same face should form a low rank matrix (ideally rank one), so they define the low rank loss based on the second singular value. Similarly, Zhu et al. [31] proposed a low-rank loss for 3D shape co-segmentation. The low-rank loss in AdaCoSeg measures the geometric similarity of the same semantic part across different shapes. The network is trained to minimize the rank; no actual low-rank decomposition is conducted. Our low-rank loss is fundamentally different from the one used

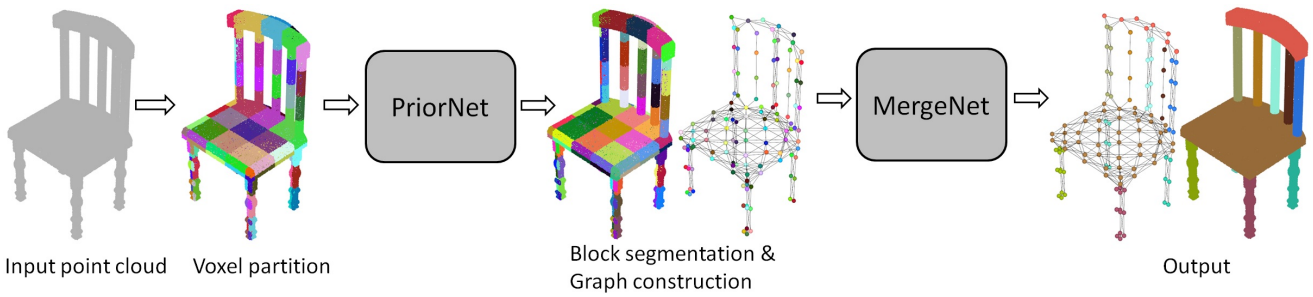


Figure 3: Pipeline overview.

in [27, 31]. our network exploits the low-rank and symmetric property of similarity matrix for point clustering, it learns to perform symmetric low-rank decomposition which directly leads to point cloud segmentation. To the best of our knowledge, our work is the first that defines a loss based on low-rank decomposition in training a point cloud segmentation network.

Graph Neural Networks (GNNs). Over past a few years, a series of graph neural networks and subsequent variants achieved promising results in various applications [30]. Landrieu et al. [8] propose a graph convolutional network-based framework to tackle the semantic segmentation of large-scale point clouds. This work is partly related to us, however, there are some important differences: 1), like other semantic segmentation frameworks, this approach [8] is classification-based and difficult to extend to our task. 2), our method learns part priors directly from a dataset through minimizing a novel low rank loss, not based on hand-crafted features. 3), our method assigns nodes based on clustering. We have no a softmax layer as the output layer for classification.

3. Our Approach

Figure 3 gives an overview of our method pipeline. Given a 3D shape represented by densely sampled point cloud, we first perform a volumetric partition to split the point cloud into a number of blocks. Each block is segmented with the PriorNet (Section 3.1). The segments of all blocks are then merged with MergeNet to form the final segmentation (Section 3.2).

3.1. Part Prior Network (PriorNet)

Given a 3D shape, the PriorNet is learned to delineate its meaningful parts (Figure 4). When the input is a partial shape (e.g., a partition block of the full shape), the PriorNet would segment it into patches which is either an independent part or a part of it. Ideally, PriorNet should avoid under-segmentation (i.e. grouping points belonging to two different parts) as much as possible. PriorNet is trained to minimize a multi-task loss function defined for each block: $L = L_{\text{sim}} + L_{\text{low-rank}}$.

Volumetric partition and block re-sampling. Since it is difficult to directly use existing neural networks to process large point clouds, we perform a volumetric partition over the point clouds into blocks and then feed each block to the PriorNet. In our implementation, the volume resolution is $7 \times 7 \times 7$. After partitioning, empty blocks are removed. Since the number of points in each block varies a lot, to facilitate training, we re-sample the point set in each block into $D = 512$ points, using farthest point sampling (FPS).

PriorNet architecture. In PriorNet, we use PointNet++ [16] as the backbone network to extract features for each point, using the default hyper-parameters in the original work. Based on the point features, we define for each block a similarity matrix to describe for any two points in the block how probable they belong to the same part. This similarity matrix possesses low rank property with the rank equal to the number of fine-grained parts in that block. Therefore, part segmentation can be obtained by minimizing the rank of the matrix.

Similarity loss. For the purpose of point cloud segmentation, we aim to learn point features so that any two points belonging to the same part are as similar as possible, while those in different parts are as dissimilar as possible. To this end, we defined a similarity matrix $S \in N_d \times N_d$ for the points in a black, where N_d is the number of points per block. To estimate S, we design a similarity loss:

$$L_{\text{sim}} = \sum_{i}^{N_d} \sum_{j}^{N_d} l_{i,j}, \tag{1}$$

where

$$l_{i,j} = \begin{cases} \|F(p_i) - F(p_j)\|_2, & I(i,j) = 1 \\ \max\{0, K - \|F(p_i) - F(p_j)\|_2\}, & I(i,j) = 0 \end{cases}$$

in which $I(i,j)$ indicates whether p_i and p_j belong to the same part in the ground-truth. F is point feature extracted

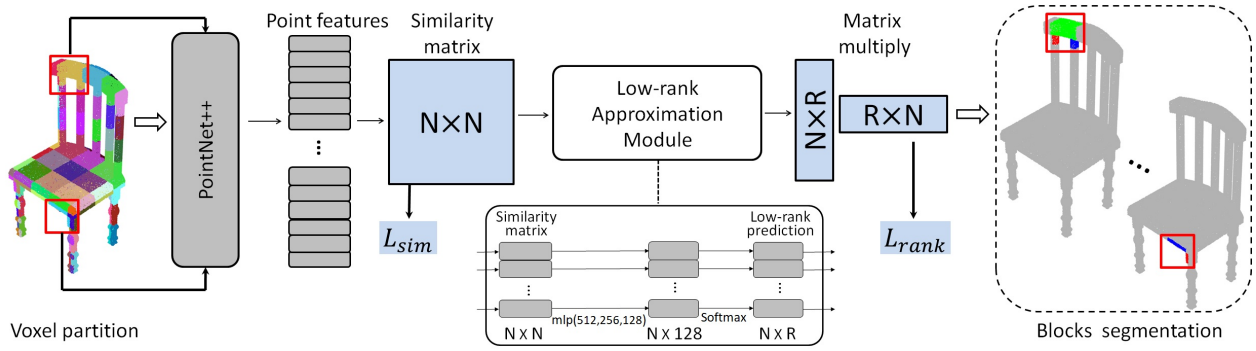


Figure 4: Network architecture of the PriorNet. After volumetric partitioning, each block is fed into the backbone network (PointNet++ [16]) to extract point-wise features. Two loss functions, similarity loss and low-rank loss, are devised to predict the segmentation result for the block.

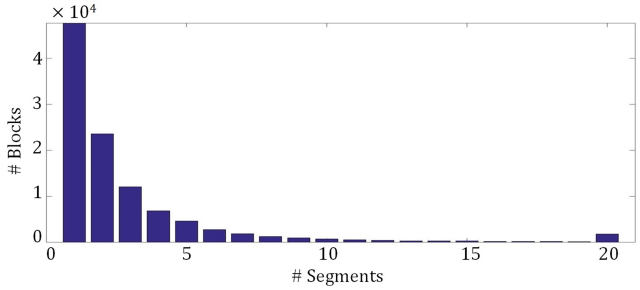


Figure 5: Statistics on segment count in a block over all the models in our dataset.

by PointNet++ [16]. K controls the minimum dissimilarity between points in different parts; we use $K = 100$ by default. We will show with experiment that our method is insensitive about the selection of K.

Remarks on the low-rank property of the similarity matrix. The "similarity" here measures how probable two given points belong to the same part. Ideally, the similarity matrix S is defined as an $N \times N$ matrix (N is point count) with each entry $s_{i,j} = I(i,j)$ being an indicator of whether point p_i and p_j belong to the same part. It is obvious that the relation "p_i and p_j belong to the same part" is an equivalence relation. This means that the corresponding relation graph has the following structure: the number of its connected components equals the number of parts, and each such connected component is a clique. Clearly, the similarity matrix S is the adjacency matrix of the relation graph. If we rearrange the rows and columns of S according to the parts, then S is a block-diagonal matrix in which each block consists of only ones, and zeros appear everywhere outside the blocks. Each block corresponds to one part. This block-diagonal matrix is obviously low-rank.

In reality, however, S may not be a clean low rank matrix. It can be "contaminated" due to noisy relations, becoming a probabilistic relation matrix with increased matrix rank. Therefore, we design a low-rank loss that optimizes to recover the ideal similarity matrix and extract fine-grained segmentation result via minimizing the low-rank loss.

Low-rank approximation of similarity matrix. Ideally, the similarity matrix S is a low rank matrix. Let us denote $S = [S_1; S_2; \ldots; S_N]$, where S_i represents i-th row of S. If p_i and p_j belong to the same part, we have $S_i = S_j$. Assuming that all linearly independent row vectors of S are $\{S_{r_1}, S_{r_2}, \ldots, S_{r_m}\}$, it is easy to verify that these row vectors are pair-wise orthogonal. By using the elementary row transformation, S can be represented by the maximal linear independent set of its row vectors:

$$\begin{cases} S_1 = a_{1,1}S_{r_1} + a_{1,2}S_{r_2} + \ldots + a_{1,m}S_{r_m} \\ S_2 = a_{2,1}S_{r_1} + a_{2,2}S_{r_2} + \ldots + a_{2,m}S_{r_m} \\ \ldots \\ S_N = a_{N,1}S_{r_1} + a_{N,2}S_{r_2} + \ldots + a_{N,m}S_{r_m} \end{cases} \tag{2}$$

Then we can rewrite (2) as

$$\begin{aligned} S = \; & [a_{1,1}, a_{2,1}, ..., a_{N,1}]^T S_{r_1} + \ldots \\ & + [a_{1,m}, a_{2,m}, ..., a_{N,m}]^T S_{r_m}. \end{aligned} \tag{3}$$

If p_i and p_{r_1} belong to the same part, we know $S_i = S_{r_1}$. According to (2), we have $a_{i,r_1} = I(p_i, p_{r_1})$. Meanwhile, for each row vector $S_{r_1} = [S_{1,r_1}, S_{2,r_1}, ..., S_{N,r_1}]$, we have $S_{i,r_1} = I(p_i, p_{r_1})$, where S_{i,r_1} represents the i-th element of S_{r_1}. Let us denote $A_{r_1} = [a_{1,1}, a_{2,1}, ..., a_{N,1}]$, it holds that $A_{r_1} = S_{r_1}$, and similarly, $A_{r_2} = S_{r_2} \ldots, A_{r_m} = S_{r_m}$. Substituting them into (3), we obtain

$$
\begin{aligned}
S &= A_{r_1}^T S_{r_1} + A_{r_2}^T S_{r_2} + ... + A_{r_m}^T S_{r_m} \\
&= S_{r_1}^T S_{r_1} + S_{r_2}^T S_{r_2} + ... + S_{r_m}^T S_{r_m} \\
&= [S_{r_1}^T, S_{r_2}^T, ..., S_{r_m}^T]_{N\times R}[S_{r_1}^T, S_{r_2}^T, ..., S_{r_m}^T]_{N\times R}^T \\
&= M_{N\times R} M_{N\times R}^T
\end{aligned} \tag{4}
$$

which is a low rank decomposition of S, with R being the rank of S.

Low-rank approximation module. Since S is symmetric, we only need to estimate $M_{N\times R}$. To this end, we design a low-rank approximation module (Figure 4). In particular, we first use a three-layer MLP, with feature dimensions 512, 256, and 128, to transform the input similarity matrix features. Since each point belongs to only one part, we then apply a softmax layer to assign each point feature a label in $(1, \ldots, R)$. The output matrix is the predicted $M_{N\times R}$ with each column representing a part instance.

Low-rank loss. Based on the symmetric and low-rank properties of the similarity matrix, we design a low rank loss:

$$
L_{\text{low-rank}} = ||M_{N\times R} M_{N\times R}^T - S^{\text{gt}}||_2^2, \tag{5}
$$

where $M_{N\times R}$ is the predicted low rank matrix of the neural network. S^{gt} is the ground-truth similarity matrix constructed using the training shapes.

Note that for each block, one cannot directly predict $M_{N\times R}$ because the actual rank R is different from block to block. According to statistics, we find that 98% of the blocks has a segment count less than 5. Therefore, we set the maximum rank to 5 in our experiments.

In the training phase, assuming that the segment count is r in each block, we can extract the top r columns from the predicted low rank matrix $M_{N\times R}$, obtaining $M_{N\times r}$. Since each point belongs to only one part, we normalize the rows of $M_{N\times r}$.

In the testing phase, we make a prediction of the similarity matrix S^{pred} and the low rank matrix $M_{N\times R}$. To determine the segment count for each block, we first take the top r columns ($r = \{1, 2, \ldots, 5\}$) to calculate the reconstructed similarity matrix $M_{N\times r} M_{N\times r}^T$ ($M_{N\times r}$ is row normalized), then select the r which attains the minimum error between the reconstructed and the predicted similarity matrix. The error is calculated as follows:

$$
M_{N\times r} = \min_{1\le r\le 5} ||M_{N\times r} M_{N\times r}^T - S^{\text{pred}}||_2^2. \tag{6}
$$

Training data preparation. To train the PriorNet, we draw training blocks from 816 training shapes. Thanks to the block-wise training strategy, the required training shapes are greatly reduced. Since both of the two training loss are related to the number of segments in blocks, we opt to balance training data for each segment count. In particular, we randomly select 4K training blocks for each segment count as training samples, and the total number is 20K for all training blocks with segment count ranging from 1 to 5. In Figure 7, we show a few examples of block segmentation result. PriorNet is quite effective in capturing the potential fine-grained parts in a block, even for those with complicated structures. Note that PriorNet is learned in a category-agnostic manner using blocks from all categories.

3.2. Segment Merging Network (MergeNet)

Given the block segmentation results, the next step is to merge all the segments to form meaningful fine-grained parts for the whole shape. To do so, we design a Graph Neural Network (GNN) which learns feature representation capturing not only local segment geometry but also global context. A Graph Convolutional Network is a dynamic model in which the hidden representation of all nodes evolves over time. It can extract high-level node representation via message passing and produces a node-level output.

Segment graph construction. We first construct a graph $G = (\mathcal{V}, \mathcal{E})$ whose nodes are the set of all segments and edges represent the adjacency relation between segments. For each node, we compute its axis-aligned bounding box (AABB). Two nodes are neighboring if their AABB's intersect. For each node $v \in \mathcal{V}$, we denote the input feature vector by x_v and its hidden representation describing the nodes state at network layer l by h_v^l. Let us use $\mathcal{N}_v$ to denote the set of neighboring nodes of v. See Figure 6 for the visualization of a segment graph.

Graph propagation model. For each segment, we have extracted point-wise features with PriorNet in the previous stage. The features are fused together with max-pooling, serving as the initial hidden feature vector x_v of segment v to encode its local geometric information. To make reliable decision in segment merging, contextual information is required, which can be obtained by the means of graph convolutions.

Each layer of the graph neural network can be written as a non-linear function:

$$
\begin{aligned}
a_v^l &= \frac{1}{\mathcal{N}_v} \sum_{u\in\mathcal{N}_v} MLP(h_u^l), \\
h_v^{l+1} &= MLP([h_v^l, a_v^l]),
\end{aligned} \tag{7}
$$

where a_v^l denotes the aggregation of the messages that node v receives from its neighbors $\mathcal{N}_v$, for layer l. MLP represents multi-layer perceptron. When updating the hidden

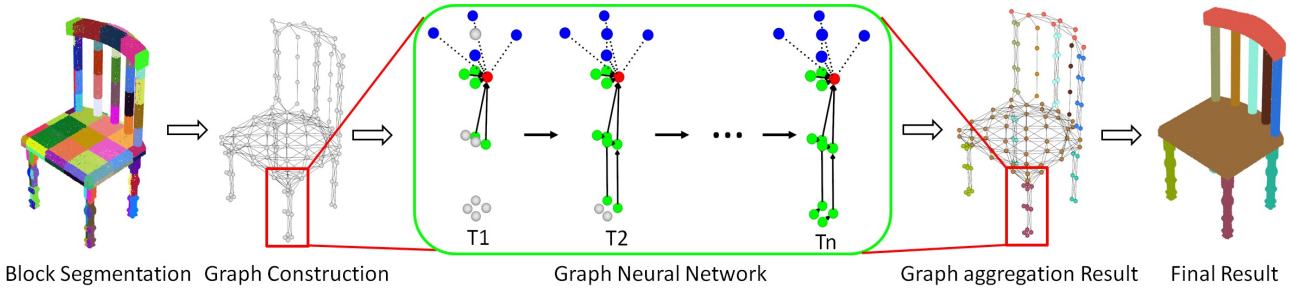


Figure 6: Network architecture of the MergeNet. Given the segmentation results of all blocks, we construct a segment graph whose nodes are segments and edges represent the adjacency relation between segments. MergeNet is a graph convolution network trained over segment graphs, aiming to merge the segments of all blocks to form the final fine-grained segmentation.

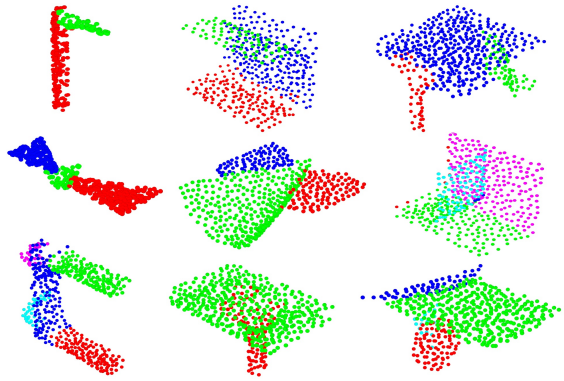


Figure 7: Some samples of block segmentation result.

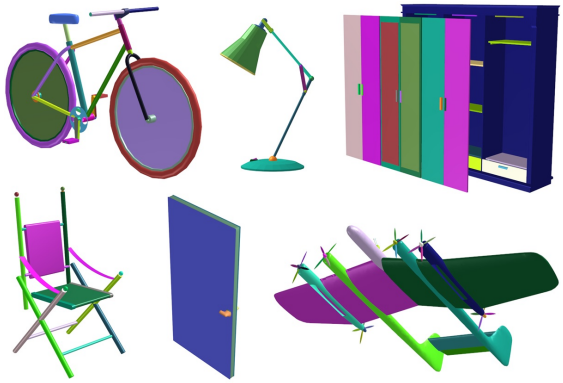


Figure 8: An overview of the fine-grained part dataset.

state, we first concatenate the hidden state h_v^l and the message a_v^l, then we feed the concatenation to an MLP. We use three layers of message passing in all our experiments.

Training loss. Similar to PriorNet, we also define a segment similarity matrix to encode whether two segments belong to the same fine-grained part. For a similar reason, this similarity matrix also possesses low rank property. For MergeNet, the parameters for the two loss functions (similarity and low rank) are the same as those for PriorNet except that the maximum rank is set to 100.

4. Results and Evaluations

Implementation details. Our network is implemented with Tensorflow. All training and testing shapes are densely sampled into 100K points to ensure that each tiny fine-grained parts are sufficiently sampled. The resolution of the volumetric partition is $7 \times 7 \times 7$. We use PointNet++ [16] as the backbone network for feature extraction with the default hyperparameters in the original paper. The batch size of PriorNet and MergeNet are 24 and 4, respectively.

Training and testing data. To facilitate quantitative evaluation, we build a challenging dataset of 3D shapes with highly fine-grained parts. These shapes were collected from the ShapeNet [3] and the PartNet [14] datasets. We choose 10 commonly seen shape categories, including 7 indoor categories and 3 outdoor ones. The topological structure of the shapes within each category is quite diverse. The second row of Table 1 reports the average number of parts per category. Note that no part labels is available in our dataset.

Timings. The training of PriorNet and MergeNet takes 17 and 5.6 hours for 100 epochs on a NVIDIA TITIAN X GPU, respectively. The testing time for each 3D point cloud is 6 seconds for PriorNet and 4 seconds for MergeNet. The segment graph construction takes about 10 seconds per shape. The total computational time is about 20 seconds per shape.

Quantitative and qualitative results. We train and test our model on our dataset, with a training/testing split of 8:2. In our method, we train one PriorNet for all categories and one MergeNet per categoery. The performance is measured by average Intersection of Union (avg. IoU) same as [19].

Table 1: Accuracy of segmentation (average Intersection of Union, in percentage) on our dataset. Row 1: The average number of fine-grained parts for each category. Row 2: Training / testing split (number of models) of our dataset. Row 3-4: Average IoU of PriorNet in different settings. Row 5-8: Average IoU of SGPN [21], GSPN [26], baseline, our method(all parts) and our method (small parts).

Rows	Bed	Chair	Clock	Door	Lamp	Table	Cabinet	Vehicle	Bicycle	Plane
1. Avg. #Parts	55.2	19.1	21.4	14.8	25.3	31.4	22.0	573.1	497.2	149.6
2. #Train / #Test	96/24	96/24	96/24	96/24	96/24	96/24	96/24	48/12	48/12	48/12
3. PriorNet (w/o low rank loss)	55.6	68.3	72.4	72.9	75.6	65.9	78.0	35.1	42.3	64.6
4. PriorNet	64.1	73.8	73.0	73.8	73.3	71.9	83.1	39.3	49.9	67.4
5. SGPN [21]	19.1	39.8	23.8	34.0	33.4	40.7	18.3	4.4	6.1	11.4
6. GSPN [26]	30.9	45.6	20.9	36.5	33.6	47.4	25.3	5.8	11.4	26.5
7. Ours (PriorNet+BL)	30.7	42.4	38.7	38.7	35.1	44.9	20.5	12.9	15.8	20.6
8. Ours (PriorNet+MergeNet)	**35.3**	**48.9**	**41.5**	**40.3**	**35.7**	**49.6**	**27.6**	**14.6**	**19.3**	**28.6**
9. Ours (Small part)	19.4	37.9	24.1	36.5	34.1	46.3	24.6	12.5	14.6	26.8

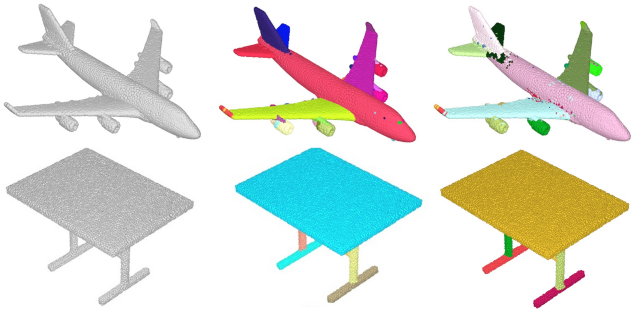


Figure 9: Some results of fine-grained segmentation.

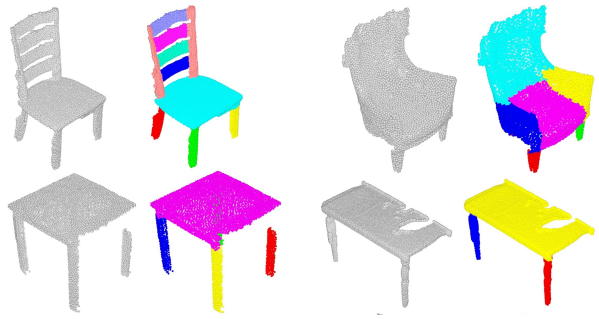


Figure 10: Segmentation results on real point cloud data by our method.

The results are reported in row 8 of Table 1. Meanwhile, in order to verify the effectiveness of our method for small part segmentation, We also have independent statistics for these small parts. The experimental results are reported in the loast row of table 1. Note that in this paper, we cannot calculate the IOU directly since there is no part label. Therefore, we design a simple strategy to calculate the IoU. For each segmented fine-grained part, we first an IoU for each part of the GT segmentation. The maximum IoU is then taken as the part IoU. Finally, we take the average IoU over all segmented parts. This is a well-practiced scheme for estimating IoUs in segmentation without part tags (e.g. [22]). In Figure 9, we show visually the fine-grained segmentation results. We also test our method on real scan data (see Figure 10).

Comparison with the state of the arts. We compare our approach with SGPN [21] and GSPN [26] , both of which are state-of-the-arts instance segmentation methods for 3D point clouds. Their tasks are per-point labeling for segmentation. To make a fair comparison, we made a simple modification to SGPN [21] to fit our task, which was to remove the semantic loss from the similarity matrix optimization. For GSPN [26], since our dataset does not contain semantic labels, we removed the semantic features from the feature backbone of R-PointNet and classification branch from the training tasks.

We report per-category IoU percentage on our dataset, see Row 5 and 6 of Table 1. The results demonstrate the significant advantage of our method with more accurate segmentation. There are two main reasons. *First*, since the number of fine-grained parts per model can be very large, a holistic segmentation network must adopt a highly powerful backbone network to extract robust and discriminant per-point feature targeting a large number of labels. Our approach adopts a divide-and-conquer scheme to overcome this difficulty. In our method, each block is segmented into a much smaller number of segments, which greatly reduces the difficulty in discriminant feature extraction. *Second*, the part count varies a lot and no part label is available in our dataset, making it difficult to compute segmentation directly from the predicted similarity matrix [21] or semantic proposals [26]. Our method achieves a robust segmentation prediction by optimizing the low rank loss.

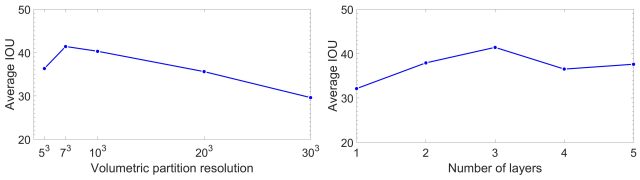


Figure 11: Segmentation accuracy (average IoU) over volumetric partition resolution (left) and the number of layers in the MergeNet (right).

Comparison with edge classification. To verify the effectiveness of our MergeNet, we design a baseline network which classifies an edge of the segment graph to determine whether the involved two adjacent segments belong to the same part. Taking a pair of adjacent segments as input, the network is trained to predict a score indicating whether the two segments belong to the same part. We then employ a hierarchical aggregation algorithm to generate the final segmentation based on the predicted scores, similar to [23]. The results are shown in Table 1 (Row 7), which are inferior to those of our method. The main reason is that the message passing process in the GCN aggregates global contextual information in the segment merging process, in contrast to the local prediction in the baseline method.

Effect of low rank loss. To evaluate the effectiveness of our low rank loss, we experiment an ablated version of PriorNet which disables the low rank loss while keeping all other parameters unchanged. The experimental results are reported in Row 3 of Table 1. For all categories, our full method with low rank loss works better. This is because the similarity matrix is usually noisy, the low rank constraint can be used to remove the noise effectively, thus improving the quality of segmentation.

Volumetric partition strategy. We evaluate the effect of the resolution of volumetric partition on segmentation quality. We experiment with the resolutions of 5^3, 7^3, 10^3, 20^3 and 30^3, while keeping all other parameters the same. In Figure 11 (left), we plot the segmentation performance (Average IoU) over different resolution settings. The results are obtained by testing on all categories and taking the average. The plot shows that best result is obtained at 7^3.

Effect of layer count. To verify the design choice of the MergeNet, we experiment with different number of network layers. Figure 11 (middle) shows the plot of average IoU over the number of layers. The results are obtained by testing on all categories and taking the average. The performance gradually improves as the number of layers increases, and then begins to oscillate. We attribute this oscillation to overfitting. We found that the best performance is obtained at $L = 3$.

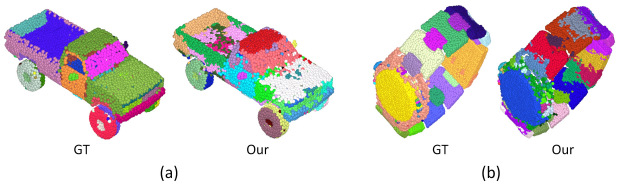


Figure 12: Failure cases caused by setting a too large max-rank.

Table 2: Comparison of AP between our method and two state-of-the-art fine-grained segmentation methods, i.e. PartNet [28] and SGPN [21].

Category	Aero	Bike	Chair	Helicopter	Sofa	Table
PartNet	**88.4**	**97.6**	**84.2**	69.4	55.8	63.2
SGPN	56.7	63.7	54.6	38.9	29.5	38.4
Ours	85.1	87.5	80.3	**73.6**	**58.7**	**71.6**

5. Conclusion

We have presented a deep-clustering-based approach to the fine-grained segmentation of 3D point clouds. The key idea is to learn geometric part priors which describe what constitutes a fine-grained part, from a dataset with fine-grained segmentation but no part semantic tags. To handle large-scale point clouds, we adopt a divide-and-conquer scheme and split the input point cloud into a set of blocks. We train a deep neural network, PriorNet, to segment each block and then merge the segments into complete fine-grained parts using a graph neural network, MergeNet.

Our method has a few limitations: First, our method requires to set the parameter of maximum rank in low-rank approximation module. However, a too large rank would cause noisy decomposition of the similarity matrix. Figure 12 shows two examples of the typical failure case could be inferior segmentation results caused by setting a too large max-rank. Second, our method is not designed to be end-to-end trainable due to the adoption of the divide-and-conquer scheme. Third, the construction of segment graphs is time consuming. In future, besides improving over the above limitations, we would like to consider extending our method to handle 3D scene point clouds, exploiting the advantage of our method in segmenting objects with significantly varying scales.

Acknowledgement

We thank the anonymous reviewers for their valuable comments. This work was supported in part by National Key Research and Development Program of China (2018AAA0102200, 2019YFF0302902), and National Natural Science Foundation of China (61932003, 61532003).

References

[1] J. Aleotti and S. Caselli. A 3d shape segmentation approach for robot grasping by parts. *Robotics and Autonomous Systems*, 60(3):358–366, 2012. 1

[2] M. M. Bronstein and I. Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1704–1711. IEEE, 2010. 2

[3] A. X. Chang, T. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 1, 6

[4] C. Chen, C. Wei, and Y. F. Wang. Low-rank matrix recovery with structural incoherence for robust face recognition. In *CVPR*, pages 2618–2625, 2012. 2

[5] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003. 2

[6] R. Hu, O. van Kaick, Y. Zheng, and M. Savva. Directions in shape analysis towards functionality. In *SIGGRAPH ASIA 2016 Courses*, page 8, 2016. 1

[7] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):433–449, 1999. 2

[8] L. Landrieu and M. Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4558–4567, 2018. 3

[9] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen. Pointcnn: Convolution on x-transformed points. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 820–830. Curran Associates, Inc., 2018. 2

[10] H. Ling and D. W. Jacobs. Shape classification using the inner-distance. *IEEE transactions on pattern analysis and machine intelligence*, 29(2):286–299, 2007. 2

[11] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, Jan 2013. 2

[12] T. Luo, K. Mo, Z. Huang, J. Xu, S. Hu, L. Wang, and H. Su. Learning to group: A bottom-up framework for 3d part discovery in unseen categories. In *International Conference on Learning Representations*, 2020. 2

[13] N. Mitra, M. Wand, H. R. Zhang, D. Cohen-Or, V. Kim, and Q.-X. Huang. Structure-aware shape processing. In *SIGGRAPH Asia 2013 Courses*, page 1, 2013. 1

[14] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. *arXiv preprint arXiv:1812.02713*, 2018. 1, 2, 6

[15] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016. 2

[16] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. 2017. 2, 3, 4, 6

[17] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE International Conference on Robotics and Automation*, pages 3212–3217. IEEE, 2009. 2

[18] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3384–3391. IEEE, 2008. 2

[19] H. Su, C. R. Qi, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page to appear, 2017. 2, 6

[20] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library, 2009. 2

[21] W. Wang, R. Yu, Q. Huang, and U. Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *CVPR*, 2018. 2, 7, 8

[22] X. Wang, B. Zhou, H. Fang, X. Chen, Q. Zhao, and K. Xu. Learning to group and label fine-grained shape components. *ACM Transactions on Graphics (TOG)*, 37(6):1–14, 2018. 1, 7

[23] X. Wang, B. Zhou, H. Fang, X. Chen, Q. Zhao, and K. Xu. Learning to group and label fine-grained shape components. *ACM Transactions on Graphics (SIGGRAPH Asia 2018)*, 37(6):Article 210, 2018. 1, 8

[24] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in Neural Information Processing Systems*, pages 2080–2088. 2009. 2

[25] K. Xu, V. G. Kim, Q. Huang, N. Mitra, and E. Kalogerakis. Data-driven shape analysis and processing. In *SIGGRAPH ASIA 2016 Courses*, page 4, 2016. 1

[26] L. Yi, W. Zhao, H. Wang, M. Sung, and L. J. Guibas. Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3947–3956, 2019. 7

[27] R. Yi, C. Zhu, P. Tan, and S. Lin. Faces as lighting probes via unsupervised deep highlight extraction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 317–333, 2018. 2, 3

[28] F. Yu, K. Liu, Y. Zhang, C. Zhu, and K. Xu. Partnet: A recursive part decomposition network for fine-grained and hierarchical shape segmentation. *arXiv preprint arXiv:1903.00709*, 2019. 1, 2, 8

[29] C. Zhang, J. Liu, Q. Tian, C. Xu, H. Lu, and S. Ma. Image classification by non-negative sparse coding, low-rank and sparse decomposition. In *CVPR*, pages 1673–1680, 2011. 2

[30] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018. 3

[31] C. Zhu, K. Xu, S. Chaudhuri, L. Yi, L. J. Guibas, and H. Zhang. Adacoseg: Adaptive shape co-segmentation with group consistency loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8543–8552, 2020. 2, 3