Learning to Group: A Bottom-Up Framework for 3D Part Discovery in Unseen Categories

Anonymous authors
Paper under double-blind review

Abstract

We address the problem of learning to discover 3D parts for objects in unseen categories. Being able to learn the geometry prior of parts and transfer this prior to unseen categories pose fundamental challenges on data-driven shape segmentation approaches. Formulated as a contextual bandit problem, we propose a learning-based iterative grouping framework which learns a grouping policy to progressively merge small part proposals into bigger ones in a bottom-up fashion. At the core of our approach is to restrict the local context for extracting part-level features, which guarantees the generalizability to novel categories. On a recently proposed large-scale fine-grained 3D part dataset, PartNet, we demonstrate that our method can transfer knowledge of parts learned from 3 training categories to 21 unseen testing categories without seeing any annotated samples. Quantitative comparisons against four strong shape segmentation baselines shows that we achieve the state-of-the-art performance.

1 Introduction

Perceptual grouping has been a long-standing problem in the study of vision systems (Hoffman & Richards, 1984). The process of perceptual grouping determines which regions of the visual input belong together as parts of higher-order perceptual units. Back to the 1930s, Wertheimer (1938) listed several vital factors, such as similarity, proximity, and good continuation, which lead to visual grouping. In the era of deep learning, high-level cues can be learned from massive annotated datasets. However, even to this day, learning-based segmentation algorithms are still far inferior to human visual systems when it comes to unseen categories. In this paper, we present a new general learning-based framework for the perceptual grouping task, focusing especially on the case of 3D shape part discovery in the zero-shot learning setting.

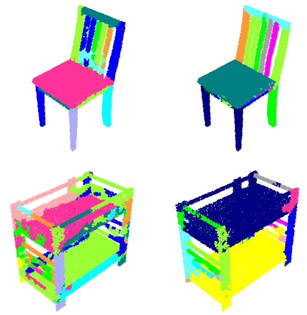


Figure 1: Shape Segmentation Results. The first row is a chair from training categories and the second row is a bed from testing categories. Left column shows PartNet-InsSeg results and right is ours.

With the power of big data, deep neural networks that learn data-driven features to segment shape parts, such as (Kalogerakis et al., 2010; Graham et al., 2018; Mo et al., 2019), have demonstrated the state-of-the-art performance on many shape segmentation benchmarks (Yi et al., 2016; Mo et al., 2019). The key to the success is to train with large-scale annotated training samples and learn to extract features that maximally exploit the data structure of the training categories. These networks usually have large receptive fields that extract features from the whole input shape and take advantage of learning global context for understanding part semantics and shape structures. While learning such features with global contextual information leads to superior performance on the training categories, they often fail miserably on unseen classes (Figure 1, row 2, column 1), as big domain gaps are observed across seen and unseen categories.

On the contrary, classical shape segmentation methods, such as (Kaick et al., 2014), that use manually designed features with relatively more local context, though giving inferior segmentation results, can perform much better for unseen object classes. In fact, many globally different shapes share similar parts locally. For example, airplanes, cars, and swivel chairs all have wheels as sub-components, even though the global geometry are totally different. Having the knowledge of wheels learned from airplanes should help recognize wheels for cars and swivel chairs, since the local geometry and semantic functionality of wheels share across the boundary of object classes. Also, automatic part discovery for unseen categories is more favorable than collecting manually labeled ground-truth for every new categories for shape segmentation.

In this paper, we aim to combine the benefits of learning data-driven part priors and extracting local-context features to perform part discovery across object categories in a zero-shot setting. We start from learning to propose a pool of superpixel-like sub-parts within local context for each shape. Then, we learn a grouping policy that seek to gradually increase the part local context and merge sub-parts for final part proposals. What lies in the heart of our algorithm is to learn a function to assess whether two parts should be merged. Different from prior deep segmentation work that learns point features, our formulation essentially learns part-level features. Borrowing ideas from Reinforcement Learning (RL), we formalize the process as a contextual bandit problem and train a local grouping policy to iteratively pick a pair of sub-parts to merge. In both steps, we learn features within part local context aiming for generalizing to unseen categories. Our *learning-based agglomerative clustering* framework deviates drastically from the prevailing deep segmentation pipelines and makes one step towards generalizable part discovery in unseen object categories.

To summarize, we make the following contributions:

- We formulate the task of zero-shot part discovery on a large-scale fine-grained shape segmentation benchmark PartNet (Mo et al., 2019);
- We propose a learning-based agglomerative clustering framework that learns to do part proposal and grouping from training categories and generalizes to unseen novel categories;
- We quantitatively compare our approach to several baseline methods and demonstrate the state-of-the-art results for part discovery in unseen object categories.

2 Related Work

Shape segmentation has been a classic and fundamental problem in computer vision and graphics. Dated back to 1990s, researchers have started to design heuristic geometric criterion for segmenting 3D meshes, including methods based on morphological watersheds (Mangan & Whitaker, 1999), K-means (Shlafman et al., 2002), core extraction (Katz et al., 2005), graph cuts (Golovinskiy & Funkhouser, 2008), random walks (Lai et al., 2008), spectral clustering (Liu & Zhang, 2004) and primitive fitting (Attene et al., 2006a), to name a few. See Attene et al. (2006b); Shamir (2008); Chen et al. (2009) for more comprehensive surveys on mesh segmentation. Many papers study mesh co-segmentation that discover consistent part segmentation over a collection of shapes (Golovinskiy & Funkhouser, 2009; Huang et al., 2011; Sidi et al., 2011; Hu et al., 2012; Wang et al., 2012; Van Kaick et al., 2013). Our approach takes point clouds as inputs as they are closer to the real-world scanners. Different from meshes, point cloud data lacks the local vertex normal and connectivity information. Kaick et al. (2014) segments point cloud shapes under the part convexity constraints. Our work learns shared part priors from training categories and thus can adapt to different segmentation granularity required by different end-stream tasks.

In recent years, with the increasing availability of annotated shape segmentation datasets (Chen et al., 2009; Yi et al., 2016; Mo et al., 2019), many supervised learning approaches succeed in refreshing the state-of-the-arts. Kalogerakis et al. (2010); Guo et al. (2015); Wang et al. (2018a) learn to label mesh faces with semantic labels defined by human. See Xu et al. (2016) for a recent survey. More recent works propose novel 3D deep network architectures segmenting shapes represented as 2D images (Kalogerakis et al., 2017), 3D voxels (Maturana & Scherer, 2015), sparse volumetric representations (Klokov & Lempitsky, 2017; Riegler et al., 2017; Wang et al., 2017; Graham et al., 2018), point clouds (Qi et al., 2017a;b; Wang et al., 2018b; Yi et al., 2019) and graph-based representations (Yi et al., 2017). These methods take advantage of sufficient training samples of seen categories and demonstrate appealing performance for shape segmentation. However, they often perform much worse when testing on unseen categories, as the networks overfit their weights to the global shape context in training categories. Our work focus on learning context-free part knowledges and perform part discovery in a zero-shot setting on novel object classes.

There are also a few relevant works trying to reduce supervisions for shape part segmentation. Makadia & Yumer (2014) learns from sparsely labeled data that only one vertex per part is given the ground-truth. Yi et al. (2016) proposes an active learning framework to propogate part labels from a selected sets of shapes with human labeling. Lv et al. (2012) proposes a semi-supervised Conditional Random Field (CRF) optimization model for mesh segmentation. Shu et al. (2016) proposes an unsupervised learning method for learning features to group superpixels on meshes. Our work processes point cloud data and focus on a zero-shot setting, while part knowledge can be learned from training categories and transferred to unseen categories.

3 Problem Formulation

We consider the task of zero-shot shape part discovery on 3D point clouds in unseen object categories. For a 3D shape S (*e.g.* a 3D chair model), we consider the point cloud $C_S = \{p_1, p_2, \cdots, p_N\}$ sampled from the surface of the 3D model. A part $P_i = \{p_{i_1}, p_{i_2}, \cdots, p_{i_t}\} \subseteq C_S$ defines a group of points that has certain interesting semantics for some specific downstream task. A set of part proposal $\mathcal{P}_S = \{P_1, P_2, \cdots, P_S\}$ comprises of several interesting part regions on S that are useful for various tasks. The task of shape part discovery on point clouds is to produce $\mathcal{P}_S^{pred}$ for each input shape point cloud C_S. Ground-truth proposal set $\mathcal{P}_S^{gt}$ is a manually labeled set of parts that are useful for some human-defined downstream tasks. A good algorithm should predict $\mathcal{P}_S^{pred}$ such that $\mathcal{P}_S^{gt} \subseteq \mathcal{P}_S^{pred}$ within an upper-bound limit of part numbers M.

A category of shapes $T = \{S_1, S_2, \cdots\}$ gathers all shapes that belong to one semantic category. For example, T_{chair} includes all chair 3D models in a dataset. Zero-shot shape part discovery considers two sets of object categories $\mathcal{T}_{train} = \{T_1, T_2, \cdots, T_u\}$ and $\mathcal{T}_{test} = \{T_{u+1}, T_{u+2}, \cdots, T_v\}$, where $T_i \cap T_j = \emptyset$ for any $i \neq j$. For each shape $S \in T \in \mathcal{T}_{train}$, a manually labeled part proposal subset $\mathcal{P}_S^{gt} \subseteq \mathcal{P}_S$ is given for algorithms to use. It provides algorithms an opportunity to develop the concept of parts in the training categories. No ground-truth part proposals are provided for shapes in testing categories $\mathcal{T}_{test}$. Algorithms are expected to predict $\mathcal{P}_S^{pred}$ for any shape $S \in T \in \mathcal{T}_{test}$.

4 Method

Our method starts with proposing a set of small superpixel-like (Ren & Malik, 2003) sub-parts of the given shape. We refer readers to Sec. B in appendix for more details. Given a set of sub-part proposals, our method iteratively groups together the sub-parts belonging to the same parts in ground-truth and produce bigger part proposals, until no sub-part can further merge each other. The remaining sub-parts in the final stage become a pool of part proposals for the input shape.

Our perceptual grouping process is a sequential decision process. We formulate the perceptual grouping process as a contextual bandit (one-step Markov Decision Process) (Langford & Zhang, 2007). We learn a policy network to select a pair of sub-parts to merge in each iteration, as shown in Alg. 1. Our policy network is composed of two sub-modules: a purity module and a rectification module. The purity module measures how likely two sub-parts belong to the same part in ground-truth after merging and the rectification module further decides the best pair to merge. Finally, we learn a termination module that judges when to stop the merging process. We describe more technical network design choices in Sec. 4.1. To train the entire pipeline, we borrow the on-policy training scheme from Reinforcement Learning (RL) to train these networks, in order to match the data distribution during training and inference stages, as described in Sec.4.2.

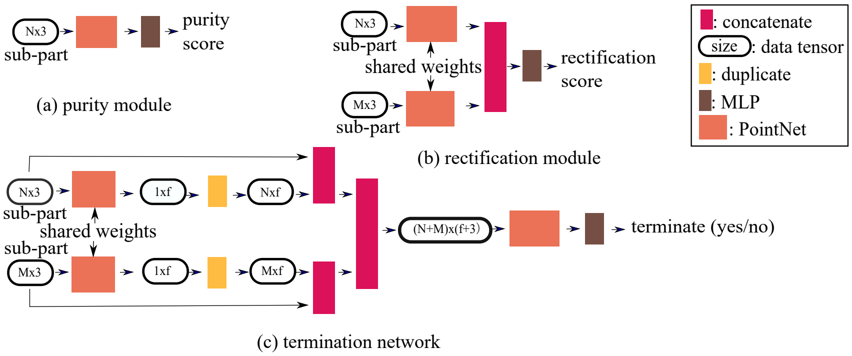
4.1 Network Designs

Our iterative grouping procedure involves two networks: a policy network picking pairs of sub-parts to group in each iteration, and a termination network controlling stopping criterion for the entire grouping process. The policy network comprises of two sub-modules: a purity module that decides if two sub-parts can merge, and a rectification module that selects the best pair to merge.

Purity Module. Two sub-parts that belong to the same ground-truth part should merge together. We define *purity score* $U(P)$ for a sub-part P as the maximum ratio of the intersection of P with

Algorithm 1 Sub-part Pair Selection and Grouping.

```
Input: A sub-parts pool $\mathcal{P} = \{P_i\}_{i \leq n}$
Input: Purity network $U$; Rectification network $R$; Termination network $V$
for $i, j \leq n$ do
    Merge two shapes: $P'_{ij} \leftarrow \{P_i \cup P_j\}$
    Calculate the purity score $u_{i,j} \leftarrow U(P'_{ij})$
    Calculate the rectification score $r_{ij} \leftarrow R(P_i, P_j)$
end for
Calculate policy $\pi(P_i, P_j) \leftarrow \frac{e^{r_{ij} u_{ij}}}{\sum_{i,j} e^{r_{ij} u_{ij}}}$
if isTraining then
    Sample pair $P_i, P_j \sim \pi(P_i, P_j)$
else
    Select the $P_i, P_j = \arg\max \pi(P_i, P_j)$
end if
if $V(P_i, P_j)$ is True then
    Delete $P_i, P_j$ from the pool
    Add $P'_{ij}$ into the pool
end if
```



(a) purity module

(b) rectification module

(c) termination network

Figure 2: Network Architectures for Three Network Modules.

the ground-truth parts $\{P_i^{gt}\}$. More formally,

$$U(P) = \max_{P_i^{gt}} \frac{\sum_p I[p \in P] I[p \in P_i^{gt}]}{\sum_p I[p \in P]} \tag{1}$$

where p enumerates all points in the shape point cloud and I is the indicator function.

We train a purity network to predict the purity score. It employs a PointNet that takes as input a merged sub-part $P_{ij} = P_i \cup P_j$ and predicts the purity score. Fig. 2 (a) shows the architecture.

Rectification Module. We observe that a purity network is not enough to fully decide the best pair of sub-parts to merge in practice. For example, when a big sub-part tries to merge with a small one from a different ground-truth part, the part geometry of the merging outcome is primarily dominated by the big sub-part, and thus the purity network tends to produce a high purity score, which results in an incorrect grouping decision. To address this issue, we consider learning a rectification module to correct the failure case given by the purity network.

We design the rectification module as in Fig. 2 (b). The rectification module takes two sub-parts as inputs, extracts features using a shared PointNet, concatenates the two part features and outputs a real valued *rectification score* $R(P)$, based purely on local information. Different from the purity module that takes the merging result as input, the rectification module explicitly takes two sub-parts as inputs in order to compare the two sub-part features for decision making.

Policy Network. We define *policy score* by making the product of *purity score* and *rectification score*. We define the policy $\pi(P_i, P_j | \mathcal{P})$ as a distribution over all possible pairs characterized by a

Algorithm 2 RL On-policy Training Algorithm.

```
Input: Policy network $\pi_\phi$ parameterized by $\phi$; Purity network $U_\theta$ parameterized by $\theta$; Verification network $V$
Initialize buffer $B$ and the networks
while True do
    Sample shape $S$ and its ground truth-label $gt$.
    Preprocess $S$ to get a sub-parts pool $\mathcal{P} = \{P_i\}_{i \leq n}$
    while $\exists$ Mergable sub-parts do
        Select and merge two sub-parts $P_i, P_j$ with Algorithm 1
        Store $(P_i, P_j, \mathcal{P})$ in $B$ and update sub-part pool $\mathcal{P}$
        Sample batch of data $(P_i^k, P_j^k, \mathcal{P}^k)_{k \leq N}$ from the buffer
        Set purity score $U_{gt}^k = U(P_i^k \cup P_j^k)$
        Set reward $M_{gt}^k = M(P_i^k, P_j^k)$
        Update policy network with policy gradient:
            $\nabla_\phi \approx \frac{1}{N} \sum_{k \leq N} \nabla \log \pi_\phi(P_i^k, P_j^k | \mathcal{P}^k) M_{gt}^k$
        Update purity module by minimizing the $l_2$ loss with purity score $U_{gt}^k$:
            $\mathcal{L}_{\text{purity}} = \frac{1}{N} \sum_{k \leq N} \|U_\theta(P_{ij}^k) - U_{gt}^k\|_2^2$
        Update termination network by minimizing the cross entropy loss :
            $\mathcal{L}_{\text{termination}} = \frac{1}{N} \sum_{k \leq N} M_{gt}^k \log V(P_i^k, P_j^k) + (1 - M_{gt}^k) \log \big(1 - V(P_i^k, P_j^k)\big)$
    end while
end while
```

softmax layer as shown in line 6 of Algorithm 1. The goal of the policy is to maximize the objective

$$\underset{\pi}{\text{maximize}}\ E_{a \sim \pi(P_i, P_j | \mathcal{P})} \left[\pi(a|\mathcal{P}) M(a) \right].$$

The reward, or the merge-ability score $M(P_i, P_j)$ defines whether we could merge two sub-parts P_i and P_j. To compute the reward $M(P_i, P_j)$: we first calculate the instance label of the corresponding ground-truth part for sub-parts P_i, P_j as l_i and l_j. We set $M(P_i, P_j)$ to be one if the two sub-parts have the same instance label and the purity scores of two sub-parts are greater than 0.8.

Termination Network The iterative grouping process continues to merge small sub-parts into bigger ones using the policy network described above. The entire merging process stops when there is no pair of sub-parts that can be merged. Since the policy scores sum to one over all pairs of sub-parts, there is no explicit signal from the policy network on when to stop. In consequences, we have to train a separate termination network that is specialized to determine stopping criterion.

The termination network takes a pair of shape as input and outputs values from zero to one after a Sigmoid layer. Fig. 2 (c) illustrates the network architecture: a PointNet first extracts part local feature for each sub-part, then two sub-part point clouds are augmented with the extracted part features and concatenated together to pass through another PointNet to obtain the final score. Notice that our design of termination network is a combination of purity module and rectification module. We want to extract both the input sub-part features and the part feature after merging.

4.2 Network Training

In this section, we illustrate how to train the two networks jointly as an entire pipeline. We use Reinforcement Learning (RL) on-policy training and borrow the standard RL training techniques, such as epilson-greedy exploration and replay buffer sampling. We also discuss the detailed loss designs for training the policy network and the termination network.

RL On-policy Training Borrowing ideas from the field of Reinforcement Learning (RL), we train the policy network and the termination network in an on-policy fashion. On-policy training alternates between the data sampling step and the network training step. The data sampling step fixes the network parameters and then runs the inference-time pipeline to collect the grouping trajectories including all pairs of sub-parts seen during the process and all the merging operations taken by the pipeline. The network training step uses the trajectory data collected from the data sampling step to compute losses for different network modules and performs steps of gradient descents to update the network parameters. We fully describe the on-policy training algorithm in Alg. 2.

We adapt psilon-greedy strategy (Mnih et al., 2013) into the training stage. We start from involving 80% random sampling samples during inference as selected pairs and decay the ratio with 10% step size in each epoch. We find that random actions not only improve the exploration in the action space and but also serve as the data-augmentation role. The random actions collect more samples to train the networks, which improves the transfer performance in unseen categories.

Also, purely on-policy training would drop all experience but only use the data sampled by current policy. This is not data efficient, so we borrow the idea from DQN (Mnih et al., 2013) and use replay buffer to store and utilize the experience. The replay buffer stores all the states and actions during the inference stage. When updating the policy networks, we sample a batch of transitions, *i.e.* , the merged sub-parts, and the sub-part pools when the algorithm merges the sub-parts from the replay buffer. The batch data is used to compute losses and gradients to update the two networks.

Training Losses As shown in Algorithm 2, to train the networks, we sample a batch of data $(P_i^k, P_j^k, \mathcal{P}^k)_{k \leq N}$ from the replay buffer, where P_i^k, P_j^k is the merged pair and $\mathcal{P}^k$ is the corresponding sub-parts pool. We first calculate the reward M_{gt}^k and ground-truth purity score U_{gt}^k for each data in the batch. For updating the rectification module, we fix the purity module and calculate the policy gradient (Sutton et al., 2000) of the policy network with the reward M_{gt}^k shown in line 11. As the rectification module is a part of the policy network, the gradient will update the rectification module by backpropogation. We then use the l_2 loss in line 12 to train the purity module and use the cross entropy loss in line 13 to train the termination network.

5 Experiments and Analysis

In this section, we conduct quantitative evaluations of our proposed framework and present extensive comparisons to four previous state-of-the-art shape segmentation methods using PartNet dataset (Mo et al., 2019) in zero-shot part discovery setting. We also show diagnostic analysis on how the discovered part knowledge transfers across different object categories.

5.1 Dataset and Evaluation

We use the recently proposed PartNet dataset (Mo et al., 2019) as the main testbed. PartNet provides fine-grained, hierarchical and instance-level part annotations for 26,671 3D models from 24 object categories. PartNet defines up to three levels of non-overlapping part segmentation for each object category, from coarse-grained parts (*e.g.* chair back, chair base) to fine-grained ones (*e.g.* chair back vertical bar, swivel chair wheel). Unless otherwise noticed, we use 3 categories (*i.e.* chairs, storage furnitures and lamps)[1] for training and take the rest 21 categories as unseen categories for testing.

In zero-shot part discovery setting, we aim to propose parts that are useful under various different use cases. PartNet provides multi-level human-defined semantic parts which can serve as a sub-sampled pool of interesting parts. Thus, we adopt Mean Recall (Hosang et al., 2015; Sung et al., 2018) as the evaluation metric to measure how predicted part pool covers the PartNet-defined parts. To elaborate the calculation of Mean Recall, we first define R_t as the fraction of ground-truth parts that have Intersection-over-Union (IoU) over t with any predicted part. Mean Recall is then defined as the average values of R_t's where t varies from 0.5 to 0.95 with 0.05 as a step size.

	🛍	🛏	🍼	🥣	🕒		🖥	🚪	🎧	🚰	👒
P	18.2	9.7	40.7	73.5	30.3	30.4	43.6	32.1	16.5	16.6	52.5
S	21.4	7	46.7	53.3	27.7	8.7	34.8	28.9	25.5	20.0	37
G	34.4	8.4	46.9	72.8	42.3	**40.6**	57.8	37.4	28.4	25.3	31.7
W	**43.1**	8.73	59.1	69.8	38.7	27.5	64.1	32.3	**38.0**	**47.8**	53.4
O	38.1	**15.2**	**52.8**	**77.4**	**43.9**	36.6	**68.7**	**40.5**	25.3	31.2	**54.3**
	⌨	🔪	💻		☕		✂		🗑	🏺	NAvg.
P	0.4	33.6	82.1	29.6	33.0	25.0	0.8	38.9	13.6	36.8	29.5
S	0.4	31	67.3	7.2	13.3	5.9	6.4	34.8	7.75	27.5	26.3
G	0.4	18.9	92.9	**39.2**	40.6	26.4	3.7	34.6	12.7	41.4	27.9
W	0.3	**62.4**	62.1	30.0	52.6	19.9	**46.3**	30.2	**23.3**	33.0	27.1
O	0.4	26.9	**96.7**	37.8	**53.3**	**30.0**	8.5	**43.3**	15.2	**42.9**	**35.1**

Table 1: Quantitative Evaluation. Algorithm P, S, G, W, O refer to PartNet-InsSeg, SGPN, GSPN, WCSeg and Ours, respectively. The number is the average among mean recall of three levels segmentation results in PartNet. We put short lines for the levels that are not defined. NAvg. is average among novel categories over shape numbers.

5.2 Baseline Methods

We compare our approach to four previous state-of-the-art methods as follows:

- **PartNet-InsSeg**: Mo et al. (2019) proposed a part instance segmentation network that employs a PointNet++ (Qi et al., 2017b) as the backbone that takes as input the whole shape point cloud and directly predicts 200 part instance masks. The method is a top-down label-prediction method that uses the global shape information.
- **SGPN**: Wang et al. (2018b) presented a learning-based bottom-up grouping method, which learns to extract per-point features and compute pairwise affinity matrix for point grouping. The method also uses PointNet++ features with global shape context.
- **GSPN**: Yi et al. (2019) introduced a deep region-based method that learns generative models for part proposals. The method proposes local part bounding boxes but still uses globally-aware PointNet++ features for predicting part masks inside boxes.
- **WCSeg**: Kaick et al. (2014) is a non-learning based method based on the convexity assumption of parts. The method leverages hand-engineered heuristics to segment shapes, and thus is agnostic to the boundary of object categories.

All the three deep learning based methods take advantage of the global shape context to achieve state-of-the-art shape part segmentation results on PartNet. However, these networks are prone to over-fitting to training categories and have a hard time transferring part knowledge to unseen categories. WCSeg, as a non-learning based method, demonstrates good generalization capability to unseen categories, but is limited by the part convexity assumption.

5.3 Results and Analysis

We compare our proposed framework to the four baseline methods under the Mean Recall metric. There are up to three levels of semantic part segmentation for each object category in PartNet. Since the segmentation levels for different categories may not share consistent part granularity (*e.g.* display level-2 parts may correspond to chair level-3 parts), we have to gather together the part proposals predicted by networks at all three levels as a joint pool of proposals for evaluation on levels of unseen categories. Thus, for PartNet-InsSeg, SGPN, GSPN and our method, we train three networks corresponding to three levels of segmentation for training categories (*e.g.* chairs, storage furnitures and lamps). We remove the part semantics prediction branch from the three baseline methods for fair comparison to our method, as semantics are not transferable to novel testing categories. For WCSeg, point normals are required by the routine to check local patch continuity. PartNet experiments (Mo et al., 2019) usu-

[1] We pick the three categories because they are big categories with several thousands models per category and provide a large variation of parts for learning.

ally assume no such point normals as inputs. Thus, we approximately compute normals based on the input point clouds by reconstructing surface with ball pivoting (Bernardini et al., 1999). Then, to obtain three-levels of part proposals for WCSeg, we manually tune hyperparameters in the procedure at each level of part annotations on training categories to match the average part prediction counts per level to the ground-truth counts in PartNet.

We present quantitative and qualitative evaluations to baseline methods in Table 1 and Figure 3. For each testing category, we report the average values of Mean Recall scores at all levels. See the appendix Table 3 for detailed numbers at all levels. We observe that our approach achieves the best performance on average among all testing novel categories, while championing 12 out of 21 categories.

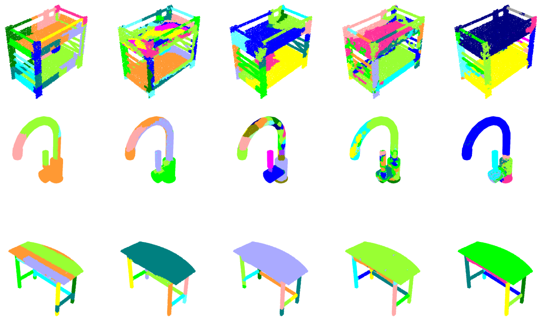


Figure 3: show qualitative results for 5 methods. From left to right, we show GSPN, SGPN, WCSeg, PartNet-InsSeg and ours performance on three testing unseen categories.

5.4 Part Knowledge Transfer Analysis

The core of our method is to learn local-context part knowledge from training categories that is able to transfer to novel unseen categories. Such learned part knowledge may also include non-transferable category-specific information, such as average size of parts, the part geometry, and the part boundary types. Training our framework on more various object categories is beneficial to learn more generalizable knowledge that shares in common. However, due to the difficulties in acquiring human annotated fine-grained parts (*e.g.* PartNet (Mo et al., 2019)), we can often conduct training on a few training categories. Thus, we are interested to know how to select categories to achieve the best performance on all categories.

	Train Category		
	🪑	⊓	💡
🪑	37.1	23.7	8.3
⊓	32.6	33.5	8.8
💡	30.9	18.8	33.4

Table 2: Cross-validation experiments for analyzing how part knowledge transfers across category boundaries.

Different object categories have different part patterns that block part knowledge transfers across category boundaries. However, presumably, similar categories, such as tables and chairs, often share common part patterns that are easier to transfer. For example, tables and chairs are both composed of legs, surfaces, bar stretchers and wheels, which offers a good opportunity for transferring local-context part knowledge. We analyze the capability of transferring part knowledge across category boundaries under our framework. Table 2 presents experimental results of doing cross-validation using chairs, tables and lamps by training on one category and testing on another. We observe that, chairs and tables transfer part knowledge to each other as expected, while the network trained on lamps demonstrates much worse performance on generalizing to chairs and tables.

6 Conclusion

In this paper, we introduced an data-driven iterative perceptual grouping pipeline for the task of zero-shot 3D shape part discovery. At the core of our method is to learn part-level features within part local contexts, in order to generalize the part discovery process to unseen novel categories. We conducted extensive evaluation and analysis of our method and presented thorough quantitative comparisons to four state-of-the-art shape segmentation algorithms. We demonstrated that our method successfully extracts locally-aware part knowledge from training categories and transfers the knowledge to unseen novel categories. Our method achieved the best performance over all four baseline methods on the PartNet dataset.

References

Marco Attene, Bianca Falcidieno, and Michela Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer*, 22(3):181–193, 2006a.

Marco Attene, Sagi Katz, Michela Mortara, Giuseppe Patané, Michela Spagnuolo, and Ayellet Tal. Mesh segmentation-a comparative study. In *IEEE International Conference on Shape Modeling and Applications 2006 (SMI'06)*, pp. 7–7. IEEE, 2006b.

Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999.

Xiaobai Chen, Aleksey Golovinskiy, and Thomas Funkhouser. A benchmark for 3D mesh segmentation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), August 2009.

Aleksey Golovinskiy and Thomas Funkhouser. Randomized cuts for 3d mesh analysis. In *ACM transactions on graphics (TOG)*, volume 27, pp. 145. ACM, 2008.

Aleksey Golovinskiy and Thomas Funkhouser. Consistent segmentation of 3d models. *Computers & Graphics*, 33(3):262–269, 2009.

Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9224–9232, 2018.

Kan Guo, Dongqing Zou, and Xiaowu Chen. 3d mesh labeling via deep convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 35(1):3, 2015.

Donald D Hoffman and Whitman A Richards. Parts of recognition. *Cognition*, 18(1-3):65–96, 1984.

Jan Hosang, Rodrigo Benenson, Piotr Dollár, and Bernt Schiele. What makes for effective detection proposals? *IEEE transactions on pattern analysis and machine intelligence*, 38(4):814–830, 2015.

Ruizhen Hu, Lubin Fan, and Ligang Liu. Co-segmentation of 3d shapes via subspace clustering. In *Computer graphics forum*, volume 31, pp. 1703–1713. Wiley Online Library, 2012.

Qixing Huang, Vladlen Koltun, and Leonidas Guibas. Joint shape segmentation with linear programming. In *ACM transactions on graphics (TOG)*, volume 30, pp. 125. ACM, 2011.

Oliver Van Kaick, Noa Fish, Yanir Kleiman, Shmuel Asafi, and Daniel Cohen-Or. Shape segmentation by approximate convexity analysis. *ACM Transactions on Graphics (TOG)*, 34(1):4, 2014.

Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3d mesh segmentation and labeling. In *ACM Transactions on Graphics (TOG)*, volume 29, pp. 102. ACM, 2010.

Evangelos Kalogerakis, Melinos Averkiou, Subhransu Maji, and Siddhartha Chaudhuri. 3d shape segmentation with projective convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3779–3788, 2017.

Sagi Katz, George Leifman, and Ayellet Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer*, 21(8-10):649–658, 2005.

Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 863–872, 2017.

Yu-Kun Lai, Shi-Min Hu, Ralph R Martin, and Paul L Rosin. Fast mesh segmentation using random walks. In *Proceedings of the 2008 ACM symposium on Solid and physical modeling*, pp. 183–191. ACM, 2008.

John Langford and Tong Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, pp. 817–824. Citeseer, 2007.

Rong Liu and Hao Zhang. Segmentation of 3d meshes through spectral clustering. In *12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings.*, pp. 298–305. IEEE, 2004.

Jiajun Lv, Xinlei Chen, Jin Huang, and Hujun Bao. Semi-supervised mesh segmentation and labeling. In *Computer Graphics Forum*, volume 31, pp. 2241–2248. Wiley Online Library, 2012.

Ameesh Makadia and Mehmet Ersin Yumer. Learning 3d part detection from sparsely labeled data. In *2014 2nd International Conference on 3D Vision*, volume 1, pp. 311–318. IEEE, 2014.

Alan P Mangan and Ross T Whitaker. Partitioning 3d surface meshes using watershed segmentation. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):308–321, 1999.

Daniel Maturana and Sebastian Scherer. 3d convolutional neural networks for landing zone detection from lidar. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3471–3478. IEEE, 2015.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 909–918, 2019.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652–660, 2017a.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems 30*. 2017b.

Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *null*, pp. 10. IEEE, 2003.

Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3577–3586, 2017.

Ariel Shamir. A survey on mesh segmentation techniques. In *Computer graphics forum*, volume 27, pp. 1539–1556. Wiley Online Library, 2008.

Shymon Shlafman, Ayellet Tal, and Sagi Katz. Metamorphosis of polyhedral surfaces using decomposition. In *Computer graphics forum*, volume 21, pp. 219–228. Wiley Online Library, 2002.

Zhenyu Shu, Chengwu Qi, Shiqing Xin, Chao Hu, Li Wang, Yu Zhang, and Ligang Liu. Unsupervised 3d shape segmentation and co-segmentation via deep learning. *Computer Aided Geometric Design*, 43:39–52, 2016.

Oana Sidi, Oliver van Kaick, Yanir Kleiman, Hao Zhang, and Daniel Cohen-Or. *Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering*, volume 30. ACM, 2011.

Minhyuk Sung, Hao Su, Ronald Yu, and Leonidas J Guibas. Deep functional dictionaries: Learning consistent semantic structures on 3d models from functions. In *Advances in Neural Information Processing Systems*, pp. 485–495, 2018.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Oliver Van Kaick, Kai Xu, Hao Zhang, Yanzhen Wang, Shuyang Sun, Ariel Shamir, and Daniel Cohen-Or. Co-hierarchical analysis of shape structures. *ACM Transactions on Graphics (TOG)*, 32(4):69, 2013.

Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36 (4):72, 2017.

Pengyu Wang, Yuan Gan, Panpan Shui, Fenggen Yu, Yan Zhang, Songle Chen, and Zhengxing Sun. 3d shape segmentation via shape fully convolutional networks. *Computers & Graphics*, 70: 128–139, 2018a.

Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018b.

Yunhai Wang, Shmulik Asafi, Oliver Van Kaick, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. Active co-analysis of a set of shapes. *ACM Transactions on Graphics (TOG)*, 31(6):165, 2012.

Max Wertheimer. Laws of organization in perceptual forms. 1938.

Kai Xu, Vladimir G Kim, Qixing Huang, Niloy Mitra, and Evangelos Kalogerakis. Data-driven shape analysis and processing. In *SIGGRAPH ASIA 2016 Courses*, pp. 4. ACM, 2016.

Li Yi, Vladimir G Kim, Duygu Ceylan, I Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, Leonidas Guibas, et al. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (TOG)*, 35(6):210, 2016.

Li Yi, Hao Su, Xingwen Guo, and Leonidas J Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2282–2290, 2017.

Li Yi, Wang Zhao, He Wang, Minhyuk Sung, and Leonidas 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*, pp. 3947–3956, 2019.

A Full Experiment Results

We present the full table including Mean Recall scores at all levels and the performance on seen categories in Table 3.

B Sub-Part Proposal Module

Given a shape represented as a point cloud, we first propose a pool of small superpixel-like (Ren & Malik, 2003) sub-parts as the building blocks. We employ furthest point sampling to sample 256 local seed points on each input shape. To capture the local part context, we extract PointNet (Qi et al., 2017a) features within a local 0.02-radius[2] neighborhood around each seed point. Then, we train a local PointNet segmentation network that takes as inputs the points within a 0.2-radius ball around every seed point and output a binary segmentation mask indicating a sub-part proposal. Finally, every sub-part proposal is assigned a partness score to get rid of sub-parts that are actually covering multiple parts in ground-truth. Figure 4 describes the process with more details.

C Ablation Study

In order to justify the proposed design of modules and training strategies, we conduct experiments to validate them and show the results in Table 4.

- **Rectification** The rectification module is involved to rectify the failure cases for purity network. Our experiments shows that without the rectification module, our decision process will easily converge to a trajectory that a pair of sub-part with in-balanced size will always be chosen to merge results in situations that one huge sub-part dominate the sub-part pool and bring in large performance drop as shown in Table 4, the ”no rectification” row.

[2] All shape point clouds are normalized into a unit-radius sphere.

	Seen Category				Novel Category								
	🪑	💡	🗄	SAvg.	👜	🛏	🍾	🥣	🕒	▤	🖥	🚪	🎧
P1	74.4	64.3	27.6	57.0	18.2	14.7	49.6	73.5	33.4	37.3	43.2	42.4	24.2
P2	40.2	23.2	54.7	39.6	-	7.8	-	-	-	32.2	-	-	-
P3	42.0	39.3	21.6	34.6	-	6.6	31.8	-	27.1	21.6	44	21.8	8.7
Avg.	**55.7**	50.5	**23.8**	**43.7**	18.2	9.7	40.7	73.5	30.3	30.4	43.6	32.1	16.5
S1	57.1	56.2	13.6	42.0	21.4	10.7	57.6	53.3	37.5	13	38.4	44.1	43.1
S2	38.2	42.1	11.4	28.2	-	6.3	-	-	-	7.9	-	23.7	-
S3	31.3	34.4	9.4	23.8	-	4	35.8	-	17.9	5.2	31.2	19	7.9
Avg.	42.2	44.2	11.5	31.3	21.4	7	46.7	53.3	27.7	8.7	34.8	28.9	25.5
G1	56.7	57.8	17.7	43.4	34.4	17.2	56.0	72.8	55.6	53.5	63.7	55.6	46.9
G2	35.2	42	13.5	27.4	-	4.6	-	-	-	40.5	-	30.2	-
G3	27.1	31.2	12.1	22.1	-	3.4	37.8	-	28.9	27.8	51.9	26.5	9.9
Avg.	39.7	43.7	14.4	31.0	34.4	8.4	46.9	72.8	42.3	**40.6**	57.8	37.4	28.4
W1	29.0	55.8	5.4	24.3	43.1	13.6	70.1	69.8	48.7	42.3	64.1	44.9	50.2
W2	30.7	53.2	1.5	21.2	-	6.9	-	-	-	23.4	-	27.5	-
W3	29.2	50	1.8	21.2	-	5.7	48.1	-	28.7	16.9	58.0	24.6	25.8
Avg.	29.6	53.0	2.9	22.2	**43.1**	8.73	59.1	69.8	38.7	27.5	64.1	32.3	**38.0**
O1	46.7	63	24.9	41.3	38.1	18.2	66.7	77.4	57.0	39.4	72.1	59.5	40.1
O2	40.2	54.7	23.2	34.8	-	14.5	-	-	-	40.7	-	34.2	-
O3	35.3	43.6	21.5	31.1	-	13	38.8	-	30.8	29.7	65.3	27.9	10.5
Avg.	40.7	**53.8**	23.2	35.7	38.1	**15.2**	**52.8**	**77.4**	**43.9**	36.6	**68.7**	**40.5**	25.3

	Novel Category												
	🚰	👒	⌨	🔪	💻	▣	☕	▯	✂	╥	🗑	🏺	NAvg.
P1	19.4	52.5	0.4	43.2	82.1	42	33	31.6	0.8	56.0	23.6	38.0	31.0
P2	-	-	-	-	-	28.5	-	25.4	-	32.4	-	-	31.7
P3	13.8	-	-	23.9	-	18.3	-	18	-	28.4	3.6	35.5	25.8
Avg.	16.6	52.5	0.4	33.6	82.1	29.6	33.0	25.0	0.8	38.9	13.6	36.8	29.5
S1	23.3	37	0.4	39.3	67.3	11.1	13.3	7.5	6.4	48.2	12.7	28.6	27.1
S2	-	-	-	-	-	7.1	-	5.4	-	29.4	-	-	28.3
S3	16.6	-	-	22.7	-	3.4	-	4.9	-	26.7	2.8	26.3	23.6
Avg.	20.0	37	0.4	31	67.3	7.2	13.3	5.9	6.4	34.8	7.75	27.5	26.3
G1	32.5	31.7	0.4	25.6	92.9	62.3	40.6	41.4	3.7	49.9	23.2	42.4	2.0
G2	-	-	-	-	-	34.6	-	24.4	-	28.8	-	-	28.3
G3	18	-	-	12.2	-	20.7	-	13.4	-	25	2.2	40.3	23.5
Avg.	25.3	31.7	0.4	18.9	92.9	**39.2**	40.6	26.4	3.7	34.6	12.7	41.4	27.9
W1	46.3	53.4	0.3	67.7	62.1	51.3	52.6	38.4	46.3	38.2	23.3	34.2	29.0
W2	-	-	-	-	-	22.3	-	13.2	-	27.7	-	-	27.0
W3	49.2	-	-	57.1	-	16.5	-	8.1	-	24.6	3.0	31.7	25.3
Avg.	**47.8**	53.4	0.3	**62.4**	62.1	30.0	52.6	19.9	46.3	30.2	**23.3**	33.0	27.1
O1	35.5	54.3	0.4	32.3	96.7	55.8	53.3	40.5	8.5	57.5	25.3	44.5	**35.9**
O2	-	-	-	-	-	36.2	-	28.8	-	38.5	-	-	**37.8**
O3	26.9	-	-	21.4	-	21.3	-	20.7	-	34.0	5.0	41.2	**31.6**
Avg.	31.2	**54.3**	0.4	26.9	**96.7**	37.8	**53.3**	**30.0**	8.5	**43.3**	15.2	**42.9**	**35.1**

Table 3: Quantitative Evaluation. Algorithm P, S, G, W, O refer to PartNet-InsSeg, SGPN, GSPN, WCSeg and Ours, respectively. The number 1, 2 and 3 refer to the three levels of segmentation defined in PartNet. We put short lines for the levels that are not defined. SAvg. is average among seen categories over shape numbers, while NAvg. is average among novel categories over shape numbers.

- **On-Policy Training** The on-policy training will sample training data that matches the inference process without the requirement of carefully designing the sampling strategy. Without it, our networks suffer from slightly decrease in performance as shown in Table 4, the ”off-policy” row.

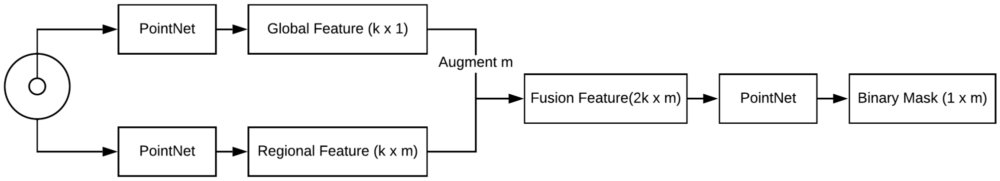


Figure 4: Learning-based sub-part proposal module.

	Test Category		
models	Chair	Bed	Faucet
no rectification	34.4	7.8	17.9
off-policy	35.3	8.5	20
full-model	36.6	9	21.2

Table 4: Ablation study