SegGroup: Seg-Level Supervision for 3D Instance and Semantic Segmentation

An Tao, *Graduate Student Member, IEEE*, Yueqi Duan, *Member, IEEE*, Yi Wei, Jiwen Lu, *Senior Member, IEEE*, and Jie Zhou, *Senior Member, IEEE*

***Abstract*— Most existing point cloud instance and semantic segmentation methods rely heavily on strong supervision signals, which require point-level labels for every point in the scene. However, such strong supervision suffers from large annotation costs, arousing the need to study efficient annotating. In this paper, we discover that the locations of instances matter for both instance and semantic 3D scene segmentation. By fully taking advantage of locations, we design a weakly-supervised point cloud segmentation method that only requires clicking on one point per instance to indicate its location for annotation. With over-segmentation for pre-processing, we extend these location annotations into segments as seg-level labels. We further design a segment grouping network (SegGroup) to generate point-level pseudo labels under seg-level labels by hierarchically grouping the unlabeled segments into the relevant nearby labeled segments, so that existing point-level supervised segmentation models can directly consume these pseudo labels for training. Experimental results show that our seg-level supervised method (SegGroup) achieves comparable results with the fully annotated point-level supervised methods. Moreover, it outperforms the recent weakly-supervised methods given a fixed annotation budget. Code is available at https://github.com/antao97/SegGroup.**

***Index Terms*— Point cloud segmentation, seg-level supervision, weakly-supervised learning, graph neural network.**

I. Introduction

RECENT years have witnessed significant progress on analyzing different 3D geometric data structures, including point cloud [1], [2], mesh [3], [4], voxel grid [5], [6], multi-view [7], [8] and implicit function [9]–[11]. Due to the popularity of varying scanning devices, 3D point cloud data is easy to obtain and thus arouses increasingly attention. Recently, many deep learning methods have been proposed to directly operate on point clouds and have achieved encouraging performance [1], [2], [12]–[15].

Manuscript received 9 July 2021; revised 22 March 2022 and 4 June 2022; accepted 1 July 2022. Date of publication 19 July 2022; date of current version 22 July 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFA0700802, in part by the National Natural Science Foundation of China under Grant 62125603 and Grant U1813218, and in part by the Beijing Academy of Artificial Intelligence (BAAI). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Adrian Munteanu. *(Corresponding author: Yueqi Duan.)*

An Tao, Yi Wei, Jiwen Lu, and Jie Zhou are with the Department of Automation, Tsinghua University, Beijing 100084, China, and also with the Beijing National Research Center for Information Science and Technology (BNRist), Beijing 100084, China (e-mail: ta19@mails.tsinghua.edu.cn; y-wei19@mails.tsinghua.edu.cn; lujiwen@tsinghua.edu.cn; jzhou@tsinghua.edu.cn).

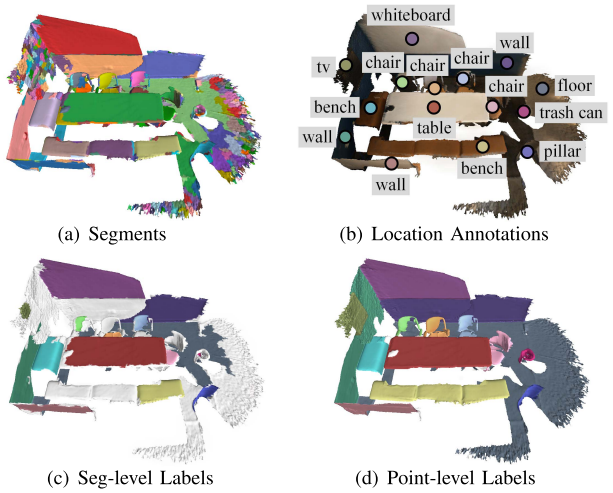
Yueqi Duan is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: duanyueqi@tsinghua.edu.cn).

Digital Object Identifier 10.1109/TIP.2022.3190709

Point cloud instance and semantic segmentation are two fundamental but challenging tasks in 3D scene understanding. Given a point cloud, instance segmentation aims to find all existing objects and mark each object with a unique instance label and a semantic class, while semantic segmentation only predicts semantic classes. Over the past few years, strong point-level supervisions that annotate every point in the scene have derived rapid performance improvement on point cloud instance and semantic segmentation tasks [16]–[21]. However, as each scene may contain a large number of points, it is highly time-consuming to annotate all the points. For example, ScanNet [22] is a widely used large-scale real-world indoor dataset. It contains 1,613 scenes and each scene has 150,000 points on average. Even though ScanNet adopts over-segmentation to reduce the annotation workload, it still needs around 22.3 minutes to annotate all the segments in one scene. With a growing number of unlabeled 3D point cloud data in real-world applications, a natural question is raised: is it necessary to label all points in a point cloud scene?

To alleviate the need for detailed point-level annotations, we focus on weak supervision in point cloud instance and semantic segmentation, which is far easier to obtain. Few prior works study weak supervision in point cloud segmentation [23]–[25], where scene-level label, subcloud-level label, and point annotation are three recent weak label forms. Scene-level labels [23] indicate the semantic classes appearing in a point cloud scene, while subcloud-level labels [23] indicate the semantic classes appearing in a spherical subcloud sampled from a point cloud. Point annotations [24]–[27] are a set of labeled points sampled from a point cloud randomly or by a specially designed algorithm. Although these weak label forms reduce annotation effort significantly, they cannot provide any instance-specific information, which makes them not suitable for point cloud instance segmentation. A recent weakly-supervised point cloud semantic segmentation method OTOC [28] uses point annotations per instance, but it cannot be directly applied to the instance segmentation task.

In this paper, we discover that *the locations of instances matter for both instance and semantic 3D scene segmentation.* Compared with the 2D images which lack depth dimension, locations of instances can be more precisely in 3D scenes. To fully take advantage of locations, we only need to click on one point per instance to indicate its location for segmentation, rather than labeling every point in the scene. With these locations, we design a weakly-supervised



(a) Segments (b) Location Annotations

(c) Seg-level Labels (d) Point-level Labels

Fig. 1. Illustration of annotation process to give instance locations. (a) Given a scene, we first perform over-segmentation to get segments as the data pre-processing step. (b) We click one point on the most representative segment of each instance to give the location and annotate it with semantic classes and instance IDs. (c) Then, we extend location annotations into segments to obtain seg-level labels. (d) Compared with point-level labels which require 22.3 minutes per scene to annotate all points, seg-level labels only require 1.93 minutes per scene to give location information of each instance.

point cloud segmentation task to explore the importance of locations.

Fig. 1 illustrates the annotation process to give instance locations. Following the labeling strategy of point-level labels in ScanNet [22], we first perform over-segmentation to obtain segments as pre-processing. Unlike 2D images which usually face occlusions and lightning variances, 3D data structures are suitable for over-segmentation for their apparent boundaries between different simple geometry parts. Then, we click one point on the most representative[1] segment of each instance to give location annotations and extend them into corresponding segments as seg-level labels. Finally, the seg-level labels are adopted as supervision signals for point cloud segmentation. According to our manual annotations, seg-level labels only require 1.93 minutes per scene to click 0.028% points compared with the strong point-level labels. However, due to the over-segmentation, these annotated segments contain 29.42% points of the whole scene.

To learn point cloud instance and semantic segmentation under seg-level supervision, we design a two-stage approach. We first propose a segment grouping network (SegGroup) to generate point-level pseudo labels from seg-level labels for the remaining unannotated points on the training set. Then, we adopt an existing point-level supervised point cloud segmentation model for standard training. The two stages are trained separately, and the evaluation of the segmentation performance is conducted on the existing point-level supervised model. In the SegGroup network, we propagate label information by grouping unlabeled segments into the relevant nearby labeled segments and conduct the grouping operation hierarchically. After all grouping operations, all points in the point cloud scene are assigned with labels that are considered as point-level pseudo labels.

Experimental results show that our seg-level supervised method (SegGroup) achieves comparable results with the fully annotated point-level supervised methods. It also outperforms the recent weakly-supervised methods [23], [25] given a fixed annotation budget. These results validate that annotating location information is a low-cost but high-yield labeling manner for 3D scene segmentation.

Our key contributions are summarized as follows:

1) We discover that the locations of instances matter for 3D scene segmentation. To fully take advantage of locations, we design a weakly-supervised point cloud segmentation method that only requires clicking on one point per instance to indicate its location for annotation.
2) We design a segment grouping network (SegGroup) to generate pseudo labels for the remaining unannotated points. Specifically, the network propagates label information by grouping unlabeled segments into the relevant nearby labeled segments hierarchically. Then, the pseudo labels are used for standard fully supervised training.
3) Experimental results show that our seg-level supervised method (SegGroup) achieves comparable results with the fully annotated point-level supervised methods. Moreover, it also outperforms the recent weakly-supervised methods given a fixed annotation budget.

II. Related Work

In this section, we briefly review two related topics: 1) point cloud segmentation, and 2) weakly-supervised image segmentation.

A. Point Cloud Segmentation

Approaches on point cloud semantic segmentation can be mainly classified into two categories: voxel-based [21], [29] and point-based [1], [2], [13], [14], [30]–[32]. Voxel-based approaches voxelized point clouds into 3D grids in order to apply powerful 3D CNNs, while point-based approaches directly design models on point clouds to learn per-point local features. Recent methods on point-based scheme include neighbouring feature pooling [33], [34], graph message passing [35], attention-based aggregation [36], [37], and kernel-based convolution [12], [14]. For point cloud instance segmentation, there are two common strategies to find instances in 3D scenes: detection-based [17], [18] and segmentation-based [16], [20], [38]. Detection-based approaches first extract 3D bounding boxes using object detection techniques, and then find the object mask inside each box. By contrast, segmentation-based approaches first predict semantic labels for each point with a semantic segmentation framework, and then group points into different objects.

While most existing methods heavily rely on strong point-level labels, only a few works have studied weakly-supervised point cloud semantic segmentation. Wei *et al.* [23] proposed scene-level labels and subcloud-level labels to indicate the semantic classes appearing in a point cloud scene and a spherical subcloud sampled from a given point cloud, respectively.

[1]In this work, we consider the largest segment of the instance as the most representative segment to indicate the instance location.

Hou *et al.* [25] proposed a 3D pre-training method that makes use of both point-level correspondences and spatial contexts in a scene with annotations of a given number of points per scene by an active selection process with the pre-trained model. Xu *et al.* [24], Hu *et al.* [26], and Zhang *et al.* [27] studied point cloud segmentation under a fraction of randomly labeled points. Although these weak label forms reduce annotation effort significantly, they cannot provide any instance-specific location information, which is a very important supervision signal in the 3D scene. In contrast, our seg-level labels are annotated per instance and are easy to be adopted by annotators when they face the need for labeling. Liu *et al.* [28] proposed a weakly-supervised point cloud semantic segmentation method OTOC by annotating one point per instance in the point cloud scene. In contrast, our method can be generally applied to semantic and instance segmentation tasks.

B. Weakly-Supervised Image Segmentation

Many works have been proposed for weakly-supervised image segmentation, where image-level supervision [39]–[43] and bounding box supervision [44]–[46] are two major lines in both instance and semantic segmentation. Image-level labels indicate the semantic classes appearing in an image, while bounding boxes further frame every instance with semantic classes. Weakly-supervised methods usually adopt a two-step process that first generates pseudo labels and then trains a supervised model treating these pseudo labels as ground truth. For image-level supervised segmentation, a common strategy is to train a classification model to recover class activation maps [47], [48]. The predicted class activation maps are then used as 'seeds' for optimization methods that grow the coarse activation maps to larger pseudo segmentation maps. Some approaches additionally employ a class-agnostic saliency estimation model [49] to capture the objectness of pixels. In contrast, bounding boxes frame every object with semantic labels, alleviating the need to estimate class activation maps. Segmentation masks can further be refined by heuristic cues [46], [50] or mean-field inference [40], [51]. The refined masks are then adopted for segmentation model training. Some works additionally use EM [40], [44], [46] for iterative refinement of the ground truth and model parameters. There are also some other weak supervision forms [52]–[54]. For example, Bearman *et al.* [52] proposed point labels to annotate each object. Lin *et al.* [53] proposed scribble labels by dragging the cursor in the center of the objects.

III. Proposed Approach

In this section, we first describe the seg-level annotations for point cloud segmentation. Then, we detail our SegGroup network to generate point-level pseudo labels from the annotated seg-level labels on the training set. Finally, we introduce the network training and implementation details.

A. Seg-Level Annotation

Following the labeling strategy of point-level labels in ScanNet [22], we follow the over-segmentation results provided by the dataset, which is obtained by employing a normal-based graph cut method [55], [56] on the mesh. The scenes on the ScanNet dataset are provided as meshes. Each scene in the ScanNet dataset is reconstructed from an RGBD video stream that records every side of a room, and the mesh structure is universal in the reconstructed scenes of other indoor datasets. The vertices in the mesh are used as the input point cloud of our SegGroup network. The results of over-segmentation remain unchanged in all the subsequent operations, including the manual annotation process and SegGroup network learning.

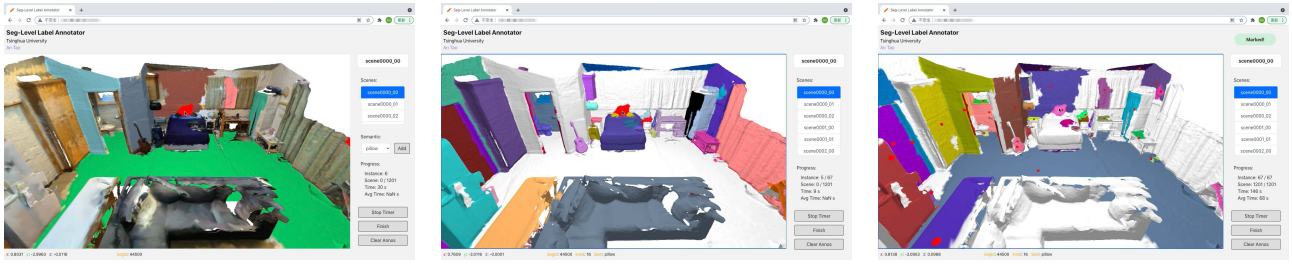
Unlike 2D images that may suffer from occlusions and lightning variances, 3D data structures usually have clear boundaries between different simple geometry parts thereby easier to perform over-segmentation. As segments are very small in this process, in most cases, each segment only contains one single object. We observe that for very few cases one segment may overlap different objects. Because the ground-truth strong labels of the ScanNet dataset are also annotated based on over-segmentation to accelerate the annotation process. Therefore, it is an intrinsic issue of this dataset that may have the minority of incorrect ground-truth labels, both in training and evaluation.

Before manual labeling started, we first generated various types of weak labels from ground-truth strong labels and conducted experiments to compare their performance in our manuscript in Table VIII. We find the performance mainly relies on the labeled segment sizes, i.e., the larger the better. According to our observation, the largest segment of each instance is usually the most central one, so we consider it can represent the location of the instance. During the annotation process, we ask the annotators to click on the point on the largest segment of the instance to show the location and consider the largest segment of an instance as the most representative segment. If it is hard to distinguish the largest segment (e.g. the sizes of some large segments for an instance are almost equal), we allow the annotators to click the point on the most central segment. With the help of over-segmentation, these location annotations are automatically extended into segments as seg-level labels.

To annotate seg-level labels, we design a WebGL annotation tool in the browser. Fig. 2 shows the interface of our annotation tool, which includes a scene display window on the left and a control panel on the right. The annotator can rotate and pan the scene to browse and annotate seg-level labels by mouse clicking. Different from point-level labels that annotate every point in the scene, we click on one point per instance to indicate its location and give its semantic class.

There are two modes to annotate the seg-level labels:

1) Annotating From Scratch: The annotation interface requires the annotator to label both the semantic class and the instance ID of the location of an instance (depicted in Fig. 2(a)). In this annotation interface, the scene is displayed with original scanned colors at the beginning. The annotator needs to choose a semantic class before annotating the location of each instance. In Fig. 2(a), the annotator is preparing to annotate the pillow.



(a) Annotating from Scratch (b) Annotating with GT Labels (c) Annotation Results

Fig. 2. The interfaces of our WebGL annotation tool. (a) We design an annotation interface that requires the annotator to label both the semantic class and the instance ID of each chosen segment. (b) Because scenes in ScanNet have ground-truth point-level labels, in this paper we choose to annotate our seg-level labels based on the ground-truth labels to reduce the annotation difficulty. (c) After annotation, the interface displays the annotation results including location annotations and seg-level labels.

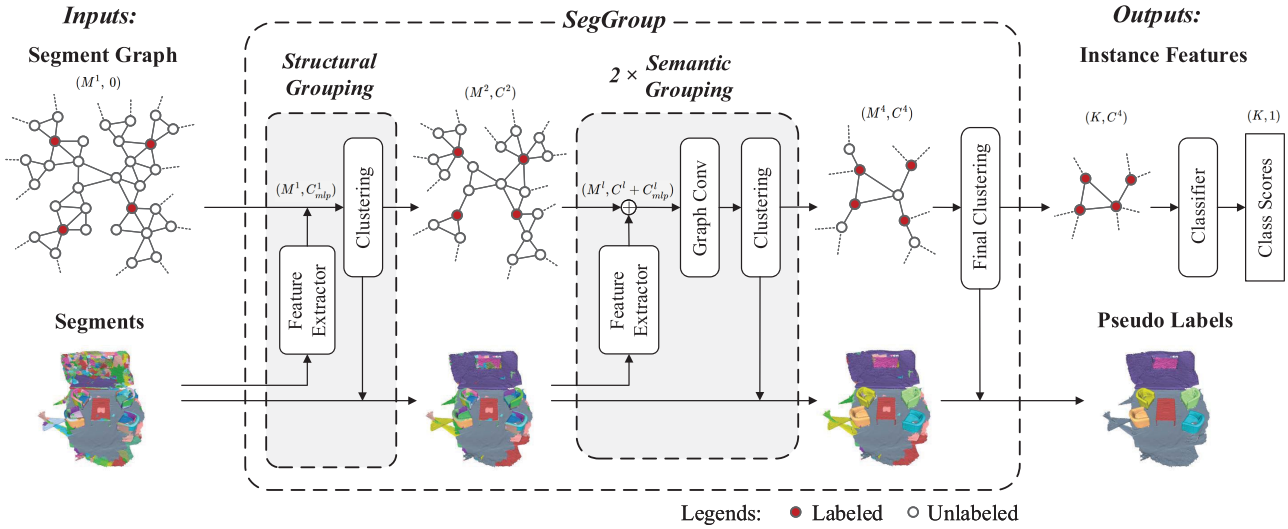


Fig. 3. The Structure of our SegGroup network. The inputs of the SegGroup network consist of a segment graph and the corresponding point cloud segments. Each node in the graph is attached with a feature vector, which describes the corresponding point cloud segment. Label information is extended from segments into nodes where the red color indicates labeled nodes. Edges in the graph denote the connected segments that are adjacent in the scene and the lengths of edges represent the similarity of segments. We design three grouping layers followed by a clustering step to group segments into instances by clustering nodes in the graph hierarchically. The classifier in the last part of the framework is used for network training.

2) Annotating With GT Labels: Because scenes in ScanNet [22] dataset have ground-truth point-level labels, in this paper we choose to annotate our seg-level labels based on the ground-truth labels to reduce the annotation difficulty (depicted in Fig. 2(b)). Compared with annotating from scratch in Fig. 2(a), in this annotation mode the annotator does not need to annotate the semantic class of each instance location.

After annotation, the WebGL annotation tool provides an interface to display the annotation results (depicted in Fig. 2(c)). The annotation results include location annotations and seg-level labels. In the display window, the positions of red balls indicate the location annotations of instances. For seg-level labels, different colors indicate they belong to different instances. The white areas of the scene are unlabeled. More details of the annotation tool can be accessed in https://github.com/antao97/SegGroup.annotator.

We calculate the average annotation time per scene based on our empirical timing results. In our work, because we only annotate the segment locations for each instance, the average instance annotation time (68s) is obtained by averaging the total annotation time for all 1,201 scenes. Our WebGL annotation tools can calculate the average annotation time per scene in real-time during the annotation process, as depicted in Figure 2(c). To simulate the standard annotation process from scratch in Fig. 2(a), we need to add the semantic annotation time. The average time to annotate a semantic class (1.5s) is obtained by testing the annotation time empirically on 10 scenes. Because the average instance number of each scene is 32, we can finally derive our total annotation time for a scene by $68 + 32 \times 1.5 = 116$s (1.93 minutes).

B. SegGroup

To learn point cloud segmentation models with seg-level labels on the training set, we propose a two-stage approach. We first design a segment grouping network (SegGroup) to generate pseudo labels for the remaining unlabeled points on the training set. Then, we adopt an existing point-level supervised point cloud segmentation model (such as PointGroup [20]) to consume the generated point-level pseudo labels for standard training and evaluate its performance on the testing set. The two stages are trained separately.

The structure of our SegGroup network is shown in Fig. 3. To fully utilize the locations of instances in 3D scene to generate pseudo labels, we assume that all segments of a single

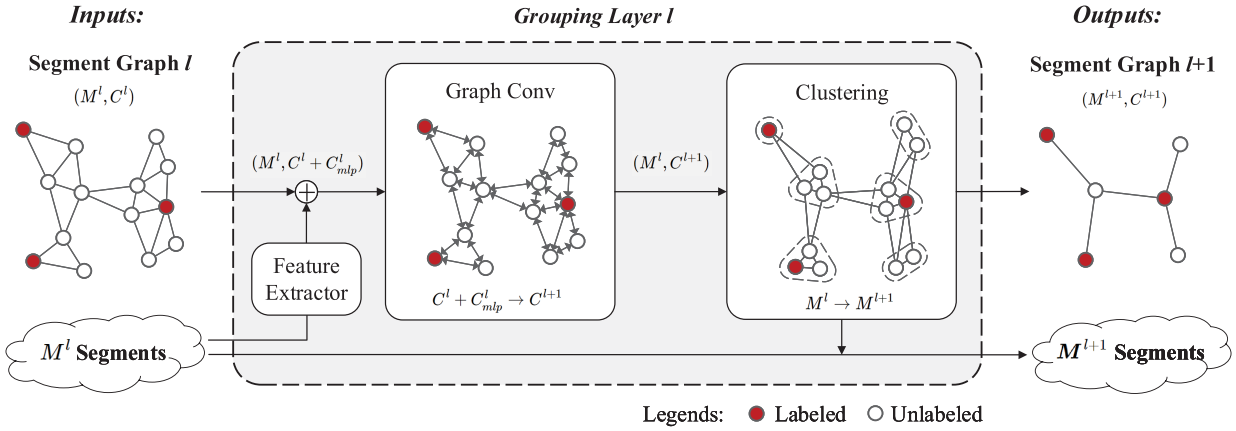


Fig. 4. The Structure of the semantic grouping layer. The inputs consist of a segment graph in layer l as well as the corresponding point cloud segments. Same as the settings in Fig. 3, each node in the graph represents a unique point cloud segment. The node features are first concatenated with segment features from point cloud segments by a feature extractor, and then updated by a graph convolution network. Finally, a node clustering algorithm groups similar nodes into new nodes. With the clustering result, similar segments are merged into large segments.

instance are interconnected so that segments can be gathered into instances according to their neighbor relationship under the guidance of instance locations. With the mesh structure of the scene data, we consider two segments to be neighbors if there exists an edge connecting two vertices belonging to two segments. Therefore, given an over-segmented point cloud scene, we build a segment graph where each node denotes a unique segment and the edge shows the adjacency between two neighboring segments in the scene. We extend the label information of each segment to the corresponding node in the graph, where the red color indicates labeled nodes. As the labeled nodes show the locations of instances, unlabeled nodes of the same instance are gradually grouped into labeled nodes through edges in the graph with our SegGroup network. The final outputs of the SegGroup are one node for each instance and all nodes are labeled with semantic classes and different instance IDs. When the nodes are merged, the segments are also merged into larger segments until all unlabeled segments are merged into nearby labeled segments. All points in the 3D scene are therefore labeled with semantic classes and instance IDs, and we take these labels as point-level pseudo labels. The additional classifier in the last part of the framework is used for network training.

We design three grouping layers followed by a final clustering process to group segments into instances by clustering nodes in the graph hierarchically. The first layer is a structural grouping layer, whose objective is to group similarly structured segments into one segment to reduce the computational costs of the subsequent grouping layers. The second and the third layer are semantic grouping layers. Because the semantic and structural grouping layers are almost the same except that the semantic grouping layer has an additional graph convolution network, we only introduce the semantic grouping layer in the following content.

The structure of the semantic grouping layer is shown in Fig. 4. The inputs of the l-th layer consist of a segment graph with M^l nodes as well as its corresponding M^l point cloud segments. Each node is represented by a C^l dimensional vector which is considered as the node feature. The output of the grouping layer is a new graph with M^{l+1} nodes as well as its corresponding M^{l+1} point cloud segments, where $M^{l+1} \leq M^l$. The node features of the output new graph are in C^{l+1} dimensional.

In the following content, we introduce the specific blocks of the semantic grouping layer.

1) Feature Extractor: We adopt a shared EdgeConv [13] network to obtain C^l_{mlp} dimensional segment features for each input point cloud segment of the grouping layer individually, which serves as a local feature learning module to extract semantic information. During the forward-propagation process of the SegGroup network, segments are gradually merged, so that the receptive field of the feature extractor in the deeper grouping layer also becomes larger to extract more macroscopic information. The input C^l dimensional node features are concatenated with their corresponding newly extracted C^l_{mlp} dimensional segment features to form new node features in $C^l + C^l_{mlp}$ dimension.

2) Graph Convolution: We update the node features with a graph convolution network (GCN) [57]. Through the neighbor relationship of nodes in the graph, the goal of GCN is to semantically narrow down the difference between nodes belonging to the same instance and extend the difference between nodes belonging to different instances. Given a node feature $\vec{h}^l_i$ and its adjacent neighbor node feature $\vec{h}^l_j$ in the l-th grouping layer, the similarity coefficient e^l_{ij} is computed according to the distance between $\vec{h}^l_i$ and $\vec{h}^l_j$ as

$$e^l_{ij} = \exp(-\lambda \|\vec{h}^l_i - \vec{h}^l_j\|_2), \tag{1}$$

where λ is a positive parameter to control the slope. A small distance indicates similar node features, which leads to a large coefficient. If $i = j$, the coefficient $e^l_{ii} = 1$. To make coefficients easily comparable across different nodes, we normalize them across all choices of j as

$$a^l_{ij} = \frac{e^l_{ij}}{e^l_{ii} + \sum_{k \in \mathcal{N}^l_i} e^l_{ik}}, \tag{2}$$

Algorithm 1 Clustering in Layer l

Input: Nodes $\{n_i^l\}_{i=1}^{M^l}$, node labels $\{y_i^l\}_{i=1}^{M^l}$, node features $\{\vec{h}_i^{l'}\}_{i=1}^{M^l}$, edges $\{o_s^l\}_{s=1}^{P^l}$, distance threshold e_τ^l.
Output: Nodes $\{n_i^{l+1}\}_{i=1}^{M^{l+1}}$, node labels $\{y_i^{l+1}\}_{i=1}^{M^{l+1}}$, node features $\{\vec{h}_i^{l+1}\}_{i=1}^{M^{l+1}}$, edges $\{o_s^{l+1}\}_{s=1}^{P^{l+1}}$.

1: **for** every node n_i^l **do**
2: $C_i^l \leftarrow \{n_i^l\}$ // initialize clusters
3: $z_i^l \leftarrow y_i^l$ // initialize cluster labels
4: **for** every edge $o_s^l = (n_a^l, n_b^l)$ **do**
5: Find $n_a^l \in C_p^l$, $n_b^l \in C_q^l$
6: **if** $C_p^l == C_q^l$ **then**
7: Continue // belong to same cluster
8: **if** $z_p^l \neq$ None and $z_q^l \neq$ None **then**
9: Continue // both clusters are labeled
10: **if** $\mathrm{dist}(\vec{h}_a^{l'}, \vec{h}_b^{l'}) < e_\tau^l$ **then**
11: $C_p^l \leftarrow C_p^l \cup C_q^l$ // merge clusters
12: Delete C_q^l
13: **if** $z_q^l \neq$ None **then**
14: $z_p^l \leftarrow z_q^l$ // update cluster label
15: Delete z_q^l
16: Obtain M^{l+1} clusters and M^{l+1} labels
17: Sort clusters and labels into $\{C_i^l\}_{i=1}^{M^{l+1}}$ and $\{z_i^l\}_{i=1}^{M^{l+1}}$
18: **for** every cluster C_i^l **do**
19: $n_i^{l+1} \leftarrow C_j^l$ // get node
20: $y_i^{l+1} \leftarrow z_i^l$ // get node label
21: $\vec{h}_i^{l+1} \leftarrow \mathrm{maxpool}(\{\vec{h}_k^{l'}\}_{n_k^l \in C_i^l})$ // get node feature
22: **for** every edge $o_s^l = (n_a^l, n_b^l)$ **do**
23: Find $n_a^l \in C_p^l$, $n_b^l \in C_q^l$
24: **if** $C_p^l \neq C_q^l$ **then**
25: $o_s^{l+1} \leftarrow (n_p^{l+1}, n_q^{l+1})$ // get edge
26: Obtain P^{l+1} edges
27: Sort edges into $\{o_s^{l+1}\}_{s=1}^{P^{l+1}}$
28: **return** $\{n_i^{l+1}\}_{i=1}^{M^{l+1}}$, $\{y_i^{l+1}\}_{i=1}^{M^{l+1}}$, $\{\vec{h}_i^{l+1}\}_{i=1}^{M^{l+1}}$, $\{o_s^{l+1}\}_{s=1}^{P^{l+1}}$ // return results for next layer

Algorithm 2 Final Clustering

Input: Nodes $\{n_i^l\}_{i=1}^{M^l}$, node labels $\{y_i^l\}_{i=1}^{M^l}$, node features $\{\vec{h}_i^l\}_{i=1}^{M^l}$, edges $\{o_s^l\}_{s=1}^{P^l}$.
Output: Nodes $\{n_i^l\}_{i=1}^{K}$, node labels $\{y_i^l\}_{i=1}^{K}$, node features $\{\vec{h}_i^l\}_{i=1}^{K}$, edges $\{o_s^l\}_{s=1}^{Q}$.

1: **while** exist $y_i^l ==$ None **do**
2: **for** every node n_i^l that $y_i^l ==$ None **do**
3: Find the neighborhood $\mathcal{N}_i^l$ of node n_i^l by edges
4: Find node $n_k^l \in \mathcal{N}_i^l$ that minimize $\mathrm{dist}(\vec{h}_i^l, \vec{h}_k^l)$
5: $n_k^l \leftarrow \{n_i^l, n_k^l\}$
6: Delete node n_i^l
7: Delete node label y_i^l
8: $\vec{h}_k^l \leftarrow \mathrm{maxpool}(\{\vec{h}_i^l, \vec{h}_k^{l'}\})$
9: Delete node feature $\vec{h}_i^l$
10: Delete edge $o_s^l = (i, k)$
11: Obtain K nodes, node labels, and node features
12: Sort into $\{n_i^l\}_{i=1}^K$, $\{y_i^l\}_{i=1}^K$, $\{\vec{h}_i^l\}_{i=1}^K$
13: Obtain Q edges
14: Sort into $\{o_s^l\}_{s=1}^{P^{l'}}$
15: **return** $\{n_i^l\}_{i=1}^K$, $\{y_i^l\}_{i=1}^K$, $\{\vec{h}_i^l\}_{i=1}^K$, $\{o_s^l\}_{s=1}^Q$

where $\mathcal{N}_i^l$ is the neighborhood of node i in the graph. The normalized similarity coefficients are used to compute a linear combination of adjacent segment features. After a shared linear transformation parametrized by a weight matrix $\mathbf{W}^l$ and a nonlinearity σ, the output node feature $\vec{h}_i^{l'}$ is finally computed as

$$\vec{h}_i^{l'} = \sigma\left(a_{ii}^l \mathbf{W}^l \vec{h}_i^l + \sum_{k \in \mathcal{N}_i^l} a_{ik}^l \mathbf{W}^l \vec{h}_k^l\right). \tag{3}$$

In grouping layer l, the dimension of $\vec{h}_i^l$ is $C^l + C_{mlp}^l$, and the dimension of $\vec{h}_i^{l'}$ is C^{l+1}.

3) Clustering: We further design a straightforward but effective node clustering algorithm to group similar neighboring nodes into one node. Given a grouping layer l, for all neighboring pairwise nodes, if they do not belong to different instances and the distance between node features is below a given threshold e_τ^l, we merge the two nodes into a new node. Finally, the algorithm produces M^{l+1} nodes from M^l nodes in grouping layer l. The node features of the new nodes are obtained by a max-pooling operation on their merged node features.

Algorithm 1 shows the detailed procedure to produce M^{l+1} nodes from M^l nodes in grouping layer l. Given a graph that has M^l nodes, we first initialize each node with a separate cluster and also assign the label information (labeled or unlabeled) to the clusters. Then, we gradually merge clusters according to conditions. More specifically, for each edge in the graph, we first check whether the connected two nodes are in the same cluster or their corresponding clusters are both labeled. If neither of the two conditions is true, the two clusters do not belong to different instances and then we check whether the distance between the two node features is smaller than a threshold e_τ^l. If the distance condition is met, we merge the two clusters and update label information. After finishing the edge traversal, we obtain M^{l+1} clusters and M^{l+1} corresponding labels. Then, we convert these clusters into new nodes and extend the label information to them. The node feature of each new node is obtained by a max-pooling operation on old node features of all nodes in each cluster. Before the max-pooling operation, the node features and node-wise distances are unchanged during the clustering process. Finally, we remove the edges inside each cluster to obtain a new graph. The node features are uniformly updated at the end of the process.

The label propagation process of our method considers the instance IDs of labeled segments by restraining the merging of two clusters with different instance IDs (lines 8-9 in Algorithm 1). There may exist issues in the boundaries between two neighbor instances of the same semantic class,

but we find this circumstance is very rare in 3D scenes. If this circumstance happens, the boundaries between the two instances are usually very clear. The segments of different instances near the boundaries cannot be easily merged due to geometry differences.

After the three grouping layers in Fig. 3, most of the nodes are merged. For the few remaining unlabeled nodes, we further perform the below node clustering algorithm to group all unlabeled nodes into nearby relevant nodes to generate final instance proposals.

4) Final Clustering: Different from the node clustering algorithm in the three grouping layers, this node clustering algorithm traverses all unlabeled nodes. This algorithm adopts a greedy strategy that obtains a locally optimal solution by merging each unlabeled node into the most similar neighbor node in the graph by comparing distances between node features. Finally, all the nodes in the graph are labeled and each instance in the 3D scene corresponds to a unique node. The point cloud segments become our propagated instance proposals which are considered as point-level pseudo labels.

Algorithm 2 shows the detailed procedure to produce K nodes from M^l nodes after all grouping layers, where K is the number of instances in the 3D scene. For each unlabeled node, this algorithm merges it into the most similar neighbor node in the graph by comparing distances between node features. At each step when two nodes are merged, the label information is updated and the new node feature is computed by a max-pooling operation on the two old node features. The corresponding edge is also removed to form a new graph. In the point cloud scene, the two corresponding segments are also merged into one. The clustering process continues until no unlabeled nodes exist.

Although Algorithm 2 depends on the order of the nodes, according to our design most of the nodes are meaningfully clustered by Algorithm 1 before the final node clustering process. The goal of the final node clustering in Algorithm 2 acts as a cleaning-up role to quickly combine the few remaining unlabeled nodes into nearby related nodes. The order of the nodes only has a minor effect in Algorithm 2. To validate the stability of our method, we randomly shuffle the order of nodes 3 times and find the floating range of the mIoU of generated pseudo labels is within 0.5% in semantic mIoU.

After the final clustering process, we can obtain point-level pseudo labels of an input scene. The nodes of the final graph in Fig. 4 have different instance IDs. In the next section, we show how to train the SegGroup network in order to get better pseudo labels.

C. Network Training

The SegGroup network outputs a graph containing K nodes, each of which is attached with a label and a node feature. Because these K nodes correspond to K different instances, we extend the label information into instances and consider the node features as instance features. To train the SegGroup network, we adopt a classifier network to obtain a score of labeled semantic class for each instance. A cross-entropy loss is further computed for backward-propagation. In essence, our network training scheme alternates between two steps.

TABLE I
THE DIMENSIONS OF THE FEATURES IN OUR ARCHITECTURE

Feature	Dimension	Feature	Dimension
C^1	0	C^1_{mlp}	128
C^2	128	C^2_{mlp}	64
C^3	192	C^3_{mlp}	64
C^4	256		

1) Forward-propagation. With the network parameters fixed, the SegGroup propagates labels to unlabeled segments as pseudo labels by clustering. The output instance features are further processed by the classifier to obtain semantic scores.
2) Backward-propagation. With the pseudo labels fixed, we use the cross-entropy loss computed by the semantic scores to optimize the parameters of the SegGroup and the classifier. As the network parameters are optimized, the instance features which are gathered by the node features in all grouping layers can capture more semantic information.

Similar ideas of this EM-like algorithm are widely-used to effectively refine the generated pseudo labels in weakly- and semi-supervised learning, such as [40], [44], [53].

Although the clustering algorithm may make mistakes and spread to the ambiguous parts with the hierarchical grouping operations, the generated pseudo labels are relatively accurate around the locations of instances that are indicated by seg-level labels. During the training process, the feature extractor and GCN can gradually learn semantically related features for each instance according to the 3D structures around the instance locations. As the features of nodes belonging to the same instance contain more semantic information and become closer in feature space, the clustering algorithm can gather them more correctly with the help of the given instance location and the SegGroup can output better pseudo labels. After such a virtuous cycle in network training, we finally obtain the well-trained pseudo labels and they are used in the strong supervised instance and semantic point cloud segmentation.

D. Implementation Details

The input of our architecture includes a point cloud scene with RGB colors and a segment graph. Each point in the point cloud scene is 6-dim (XYZ and RGB). The initial node feature C^1 in the segment graph is vacant, so the dimension is 0. The feature extractor in the first semantic layers extracts 64-dim features with one shared multi-layer perceptron (MLP) layer (64), while the extractor in the second layer adopts two shared MLP layers (64, 64) to obtain 64-dim features. After the SegGroup network, the output node feature C^4 is 256-dim. Table I lists the details of the dimensions of the features in our architecture. In all GCNs, we set the output feature dimension the same as the input dimension and $\lambda = 1/8$. We set the threshold e^l_τ for clustering as 6 and 2 for the structural grouping layer and the semantic grouping layer, respectively. We have tried many other thresholds (5 and 7 for the structural grouping layer, 1 and 3 for the semantic grouping layer) and found the floating range of the mIoU of generated pseudo

TABLE II
THE CLASS-SPECIFIC SEMANTIC IOUS (%) OF THE GENERATED PSEUDO LABELS ON SCANNET TRAINING SET

Method	Label	Anno. Time	Wall	Floor	Cab.	Bed	Chair	Sofa	Table	Door	Wind.	Bshf.	Pic.	Cntr.	Desk	Curt.	Fridg.	Shwr.	Toil.	Sink	Bath.	Ofurn.	Mean
MPRM [23]	Scene	0.25 min	47.3	41.1	10.4	43.2	25.2	43.1	21.9	9.8	12.3	45.0	9.0	13.9	21.1	40.9	1.8	29.4	14.3	9.2	39.9	10.0	24.4
MPRM [23]	Subcloud	3 min	58.0	57.3	33.2	**71.8**	50.4	**69.8**	47.9	42.1	44.9	**73.8**	28.0	21.5	49.5	**72.0**	38.8	44.1	42.4	20.0	48.7	34.4	47.4
Layer 1	Seg	1.93 min	37.4	51.5	30.5	20.3	18.7	11.3	35.0	38.5	11.6	9.2	**47.6**	36.4	22.0	5.5	25.3	14.0	13.5	21.2	21.0	20.2	24.5
Layer 2	Seg	1.93 min	65.5	81.4	44.1	35.0	31.5	22.3	49.0	61.4	36.3	17.6	43.6	49.1	36.7	56.4	41.4	69.1	22.0	27.8	25.2	37.0	42.6
Layer 3	Seg	1.93 min	67.8	80.1	50.1	37.9	43.7	32.0	51.8	64.2	43.2	27.4	43.5	49.2	39.7	61.2	49.3	70.0	31.0	36.1	29.6	48.6	47.8
Layer 4	Seg	1.93 min	67.9	80.1	50.9	38.1	51.2	33.5	54.5	64.3	43.7	28.4	44.9	49.3	40.5	61.4	51.5	70.0	36.0	42.1	30.3	51.2	49.5
SegGroup	Seg	1.93 min	**71.0**	**82.5**	**63.0**	52.3	**72.7**	61.2	**65.1**	**66.7**	**55.9**	46.3	42.7	**50.9**	**50.6**	67.9	**67.3**	**70.3**	**70.7**	**53.1**	**54.5**	**63.7**	**61.4**

labels is within 3% in semantic mIoU. This experimental result shows that our network is not very sensitive to the thresholds. The classifier obtains class scores with two fully-connected layers. We apply dropout with a keep probability of 0.5 in the classifier. All layers in the feature extractors and the classifier include LeakyReLU and batch normalization. In the network training, we use SGD [58] to optimize the SegGroup network with the learning rate as 0.1 and momentum as 0.9. We train the network for 6 epochs in a batch size of 8.

IV. EXPERIMENTS

In this section, we first evaluated the accuracy of the generated pseudo labels compared with ground truth labels. Then, we conducted experiments to show the performance of point cloud instance and semantic segmentation with our pseudo labels. Finally, we compared the annotation efficiency of different label forms with a fixed labeling time.

We adopted the ScanNet [22] dataset to conduct our experiment. ScanNet is a widely used large-scale real-world indoor 3D dataset, containing 1,201 training scenes, 312 validation scenes, and 100 hidden test scenes. The dataset has 40 object classes, and the evaluation of semantic segmentation is conducted on 20 classes. For instance segmentation, the wall and floor classes are ignored and only 18 classes are used for evaluation. We annotated seg-level labels in all the 40 classes, where we trained the SegGroup network on these labeled classes. We report the results on 18 classes for instance segmentation tasks and 20 classes for semantic segmentation tasks.

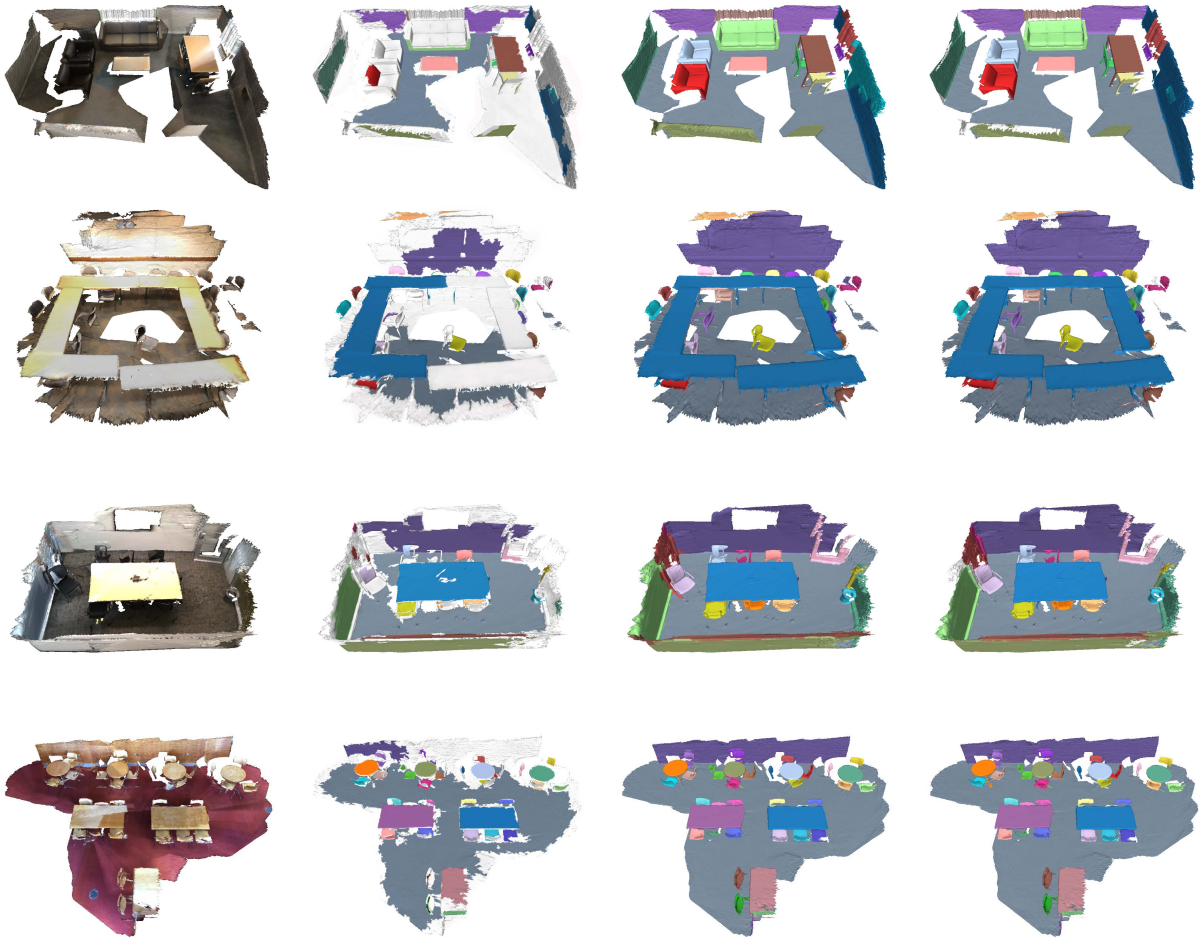
A. Pseudo Label Evaluation

We first evaluated the generated pseudo labels by instance and semantic IoUs. Table II shows the class-specific semantic IoU results of the pseudo labels on the ScanNet training set. The semantic IoUs are calculated per semantic class. We also tested the pseudo labels generated in every layer of the SegGroup network. The last row "SegGroup" indicates the final output of the network. As the label information is propagated from labeled segments into unlabeled segments with the number of grouping layers increasing, more points in the point cloud scene are annotated with the pseudo labels until all points are annotated. Experiments show that the generated pseudo labels are very close to the ground truth labels by annotating one point per instance. From the results on different classes, we observe that our method performs significantly better on the classes of *wall* and *floor* which have simple structures as well as *chair*, *sofa*, *table*, *shower*, *curtain* and *toilet* that are easy to identify and separated from a smooth surface. Table II also shows the evaluation results of the pseudo labels generated from scene-level labels and subcloud-level labels by MPRM [23], where our generated pseudo labels outperform them by a large gap. The mean IoU at Layer 1 of SegGroup is already higher than that of the scene-level labels by MPRM. Considering our method also requires less annotation time per scene compared with the subcloud-level method, experimental results show that our method has a better balance between the annotation time and the quality of the generated pseudo labels.

Besides the quantitative results that compare our generated pseudo labels with the ground truth, we also show qualitative visualizations for a more intuitive illustration in Fig. 5. On the ScanNet dataset, there are still a very small number of points remaining unlabeled for point-level annotation, so we can also observe some white details in Fig. 5(d). By only annotating one point per instance for its location, we obtain the labels for the corresponding segments with a number of points. In contrast to point-level labels, our labeling method is very cost-effective. With the benefit of seg-level labels, our SegGroup network can generate qualitative point-level pseudo labels from instance locations. From the visualization results, we observe that the pseudo labels match the ground-truth labels in almost all the areas.

B. Point Cloud Instance Segmentation

For the evaluation of the point cloud instance segmentation task, we employed PointGroup [20] to train a point cloud instance segmentation model based on the generated point-level pseudo labels. Table III shows the recent point-level and weakly-supervised point cloud instance segmentation results on the ScanNet testing set. The Point Anno. entry denotes the percentage of manually annotated point labels in the total points of the point cloud scene. Because the numbers of segment labels and point labels are the same in our method, we consider a segment label as a point label when computing the percentage of overall annotated data. AP averages the scores with IoU (Intersection over Union) threshold set from 50% to 95% with a step size of 5%, while AP 25 and AP 50 denote the AP scores with IoU threshold set as 25% and 50% respectively. We use CSC-20 and CSC-50 to denote the Contrastive Scene Contexts (CSC) [25] method that is annotated with 20 and 50 points per scene. We observe that our



(a) Point Cloud Scenes (b) Seg-level Labels (c) Pseudo Labels (d) Ground-Truth Labels

Fig. 5. The qualitative visualization results of pseudo label generation. In (b), (c) and (d), various colors indicate different (pseudo) instance labels and points in white are unlabeled. With the benefit of seg-level labels, our SegGroup network can generate qualitative point-level pseudo labels from instance locations.

TABLE III

POINT CLOUD INSTANCE SEGMENTATION RESULTS (%) ON THE SCANNET TESTING SET WITH DIFFERENT SUPERVISIONS

Method	Point Anno.	Publication	AP	AP_{50}	AP_{25}
Point-level Supervision:					
HAIS [59]	100%	ICCV'21	45.7	69.9	80.3
SSTNet [60]	100%	ICCV'21	50.6	69.8	78.9
OccuSeg [38]	100%	CVPR'20	48.6	67.2	78.8
PE [61]	100%	CVPR'21	39.6	64.5	77.6
PointGroup [20]	100%	CVPR'20	40.7	63.6	77.8
Dyco3D [62]	100%	CVPR'21	39.5	64.1	76.1
3D-MPA [63]	100%	CVPR'20	35.5	61.1	73.7
MTML [64]	100%	ICCV'19	28.2	54.9	73.1
3D-BoNet [19]	100%	NeurIPS'19	25.3	48.8	68.7
3D-SIS [18]	100%	CVPR'19	16.1	38.2	55.8
GSPN [17]	100%	CVPR'19	15.8	30.6	54.4
SGPN [16]	100%	CVPR'18	4.9	14.3	39.0
Init+Act. Point Supervision:					
CSC-20 (PointGroup) [25]	0.014%	CVPR'21	15.9	28.9	49.6
CSC-50 (PointGroup) [25]	0.034%	CVPR'21	22.9	41.4	62.0
Seg-level Supervision:					
SegGroup (PointGroup)	0.028%	-	24.6	44.5	63.7

seg-level supervised method achieves competitive results with the recent strong supervised methods. However, the seg-level labels only require 1.93 minutes to click on 0.028% of points for one scene in ScanNet on average, while the strong point-level labels need 22.3 minutes. Compared with the CSC [25] method which proposes an active labeling strategy (denoted as init+act.) to annotate points, our annotations are per instance and indicate the locations. In seg-level labels, we annotate 41.2 points on average for each 3D scene. The results show that our seg-level labels obtain much better performance than init+act. point annotations for the instance segmentation task.

TABLE IV

POINT CLOUD INSTANCE SEGMENTATION RESULTS (%) ON THE SCANNET VALIDATION SET WITH A FIXED LABELING TIME BUDGET ON THE TRAINING SET

Method	Scenes	AP	AP_{50}	AP_{25}
Point-level Supervision: PointGroup [20]	104	9.4	19.9	36.4
Seg-level Supervision: SegGroup (PointGroup)	1201	23.4	43.4	62.9

Besides the experiments above, we also compared the efficiency of different labeling strategies given a fixed annotation time budget on the training set. Table IV shows point cloud

TABLE V
POINT CLOUD SEMANTIC SEGMENTATION RESULTS (%) ON THE SCANNET TESTING SET COMPARED WITH DIFFERENT SUPERVISIONS

Method	Point Anno.	Publication	mIoU
Point-level Supervision:			
Mix3D [65]	100%	3DV'21	78.1
OccuSeg [38]	100%	CVPR'20	76.4
VMNet [66]	100%	ICCV'21	74.6
Virtual MVFusion [67]	100%	ECCV'20	74.6
MinkowskiNet [21]	100%	CVPR'19	73.4
SparseConvNet [29]	100%	CVPR'18	72.5
JSENet [68]	100%	ECCV'20	69.9
FusionNet [69]	100%	ECCV'20	68.8
KPConv [14]	100%	ICCV'19	68.4
DCM-Net [4]	100%	CVPR'20	65.8
RandLA-Net [34]	100%	CVPR'20	64.5
FusionAwareConv [70]	100%	CVPR'20	63.0
HPEIN [71]	100%	ICCV'19	61.8
SegGCN [72]	100%	CVPR'20	58.9
TextureNet [73]	100%	CVPR'19	56.6
3DMV [7]	100%	ECCV'18	48.8
PointCNN [74]	100%	NeurIPS'18	45.8
PointNet++ [2]	100%	NeurIPS'17	33.9
ScanNet [22]	100%	CVPR'17	30.6
Subcloud-level Supervision:			
MPRM (KPConv) [23]	Subcloud	CVPR'20	41.1
Rand Point Supervision:			
PSD (RandLA-Net) [27]	1%	ICCV'21	54.7
SQN (RandLA-Net) [26]	0.1%	ECCV'22	53.5
One Thing One Click:			
OTOC (SparseConvNet) [28]	0.022%	CVPR'21	69.1
Init+Act. Point Supervision:			
CSC-20 (MinkowskiNet) [25]	0.014%	CVPR'21	53.1
OTOC-20 (SparseConvNet) [28]	0.014%	CVPR'21	59.4
CSC-50 (MinkowskiNet) [25]	0.034%	CVPR'21	61.2
OTOC-50 (SparseConvNet) [28]	0.034%	CVPR'21	64.2
Seg-level Supervision:			
SegGroup (KPConv)	0.028%	-	61.1
SegGroup (MinkowskiNet)	0.028%	-	62.7

instance segmentation results under different supervisions on the ScanNet validation set with different numbers of training scenes. Given the labeling time for the whole training set (1201 scenes) with seg-level labels, only 104 scenes can be annotated with point-level labels. The results show that our seg-level labels obtain much better performance than point-level labels given the same annotation budget for the instance segmentation task.

C. Point Cloud Semantic Segmentation

For the point cloud semantic segmentation task, we trained MinkowskiNet [21] and KPConv [14] with the generated pseudo labels for evaluation. Table V shows the results of both the recent point-level and other weak supervision methods[2] on the ScanNet testing set. Following the definitions in Table III, we use OTOC-20 and OTOC-50 to denote the OTOC [28] method that is annotated with 20 and 50 points per scene. We use mIoU (mean Intersection over Union) as the evaluation metric. The results show that we achieve comparable results with the strong point-level supervised methods. Considering the huge time cost of the point-level labels, our seg-level supervised method is very competitive. Moreover, we outperform most of the weakly-supervised methods in a comparable point annotation rate. Considering the annotation time of the subcloud-level labels in MPRM [23] is about 3 minutes per scene, we also surpass MPRM on the semantic segmentation performance with a less time budget of 1.93 minutes per scene. Experimental results show that our seg-level labels provide a nice tradeoff between annotation time and segmentation accuracy, which shows that precise instance locations are crucial for 3D scene understanding.

Although the performance of OTOC [28] on semantic segmentation is better than ours, OTOC can not be directly adopted for instance segmentation. This limitation is determined by its objective function (energy function) for pseudo label generation, where both the unary term and the pairwise term only consider semantic labels. The energy function is built upon the point-level semantic prediction outputs of the semantic segmentation model. The final semantic pseudo labels are obtained by minimizing the energy function. We also find that our method uses much fewer network parameters than OTOC. When we remove the semantic segmentation model and remain the backbone for pseudo label generation, the parameter numbers are 30.11M v.s. 0.15M for OTOC and our SegGroup.

We presented the semantic IoU per class results after the classical learning process in Table VI. For the point-level and seg-level supervised methods, we observe that the rankings of class-specific IoUs from the highest to the lowest are almost the same. This phenomenon shows that our seg-level supervised method has similar behavior to the point-level supervised method. When comparing Table VI with Table II, the experimental results do not show explicit relation between the rankings of the class-specific IoUs of pseudo labels on the training set in Table II and the predicted labels on the validation set in Table VI. We think the rankings of class-specific semantic IoUs of both point-level and seg-level supervised methods in Table VI are determined by the difficulty of the classes themselves in the training process.

We also compared the efficiency of different labeling strategies on point cloud semantic segmentation following the settings in Sec. IV-B. Table VII shows the point cloud semantic segmentation results under different supervisions on the ScanNet validation set with a fixed annotation budget on the training set. Given the labeling time for the whole training set with seg-level labels, only 104 scenes can be annotated with point-level labels. Our SegGroup outperforms the fully supervised method under the same annotation cost. Combining the results in both Table IV and Table VII, our seg-level labels provide a better tradeoff between annotation time and segmentation accuracy compared with point-level methods, which shows that precise instance locations are crucial for 3D scene understanding. Annotating instance location as seg-level labels can be a low-cost but high-yield labeling manner for 3D instance and semantic segmentation.

D. Ablation Study

Before manual labeling started, we first generated various types of weak labels from ground-truth strong labels

[2]If an instance is composed of some disconnected portions, we allow the annotator to label one segment on each of them. Therefore, the point annotation rate of SegGroup is slightly higher than OTOC [28].

TABLE VI

THE CLASS-SPECIFIC SEMANTIC IOUS (%) OF POINT CLOUD SEMANTIC SEGMENTATION RESULTS ON THE SCANNET VALIDATION SET COMPARED WITH THE FULLY-SUPERVISED METHOD

Manner	Point Anno.	Wall	Floor	Cab.	Bed	Chair	Sofa	Table	Door	Wind.	Bshf.	Pic.	Cntr.	Desk	Curt.	Fridg.	Shwr.	Toil.	Sink	Bath.	Ofurn.	Mean
Point-level Supervision:																						
KPConv [14]	100%	82.1	94.4	64.4	79.5	89.2	77.7	72.3	59.1	58.0	78.7	28.5	60.6	62.0	70.6	50.8	51.1	91.7	62.6	85.9	54.0	68.7
MinkowskiNet [21]	100%	84.8	94.9	65.6	80.4	90.5	85.4	74.6	64.0	62.1	80.8	32.4	61.6	63.8	77.6	53.8	69.9	91.9	69.6	87.3	59.3	72.5
Seg-level Supervision:																						
SegGroup (KPConv)	0.028%	78.1	92.9	52.6	69.3	82.8	69.4	66.7	55.9	51.1	73.5	29.4	56.0	50.3	61.0	37.6	56.3	83.1	57.0	77.8	46.5	62.4
SegGroup (MinkowskiNet)	0.028%	78.8	93.3	53.7	73.3	83.5	74.4	67.2	55.3	49.6	72.8	27.2	57.3	54.3	63.9	43.8	66.1	87.1	58.8	80.2	49.1	64.5

TABLE VII

POINT CLOUD SEMANTIC SEGMENTATION RESULTS (%) ON THE SCANNET VALIDATION SET WITH A FIXED LABELING TIME BUDGET ON THE TRAINING SET

Method	Scenes	mIoU
Point-level Supervision:		
KPConv [14]	104	53.2
MinkowskiNet [21]	104	53.0
Seg-level Supervision:		
SegGroup (KPConv)	1201	62.4
SegGroup (MinkowskiNet)	1201	64.5

TABLE VIII

POINT CLOUD INSTANCE AND SEMANTIC SEGMENTATION RESULTS (%) ON THE SCANNET VALIDATION SET SUPERVISED BY SEG-LEVEL LABELS OBTAINED WITH DIFFERENT ANNOTATION MANNERS

Manner	Ins Seg AP	Ins Seg AP_{50}	Ins Seg AP_{25}	Sem Seg mIoU
Fully-Sup Baseline	34.8	56.9	71.3	68.7
Mechanical:				
Top-1 Segment	24.8	46.2	64.6	63.6
Top-2 Segment	25.0	43.9	62.6	64.2
Top-3 Segment	24.2	43.0	62.4	64.2
Rand Segment	20.0	36.9	57.8	46.7
Manual:				
Top-1 Segment	23.4	43.4	62.9	62.4

and conducted experiments to compare their performance. We employed PointGroup [20] and KPConv [14] to train a point cloud instance segmentation model and a point cloud semantic segmentation model based on the generated point-level pseudo labels from our SegGroup. The Mechanical part of Table VIII shows the point cloud instance and semantic segmentation results under various annotation manners on the ScanNet validation set. For mechanical seg-level labels generated from Top-N segments, we choose to annotate one point randomly in the range of Top-N largest segments for each instance. The results show that larger segments can yield better performance. We follow these experimental results to design our manual annotation rule as annotating on the largest segment of an instance as the most representative segment.

We also compared the annotation quality between manual labeling by annotators and mechanical labeling from the ground-truth point-level labels in Table VIII. Compared to mechanical annotations, the results of manual annotations are slightly worse but comparable. Therefore, our manual labeling strategy for seg-level labels is practical for future applications. Moreover, we find that for mechanical annotations the results

TABLE IX

POINT CLOUD INSTANCE AND SEMANTIC SEGMENTATION RESULTS (%) ON THE SCANNET VALIDATION SET SUPERVISED BY SEG-LEVEL LABELS OBTAINED WITH DIFFERENT ANNOTATION NUMBERS PER INSTANCE

Manner	Ins Seg AP	Ins Seg AP_{50}	Ins Seg AP_{25}	Sem Seg mIoU
Fully-Sup Baseline	34.8	56.9	71.3	68.7
Mechanical:				
One Segment	24.8	46.2	64.6	63.6
Two Segments	28.1	49.0	66.3	66.3
Three Segments	29.1	48.7	67.1	67.4
Five Segments	30.2	51.7	68.3	68.4
Seven Segments	31.4	52.5	68.6	68.5
Ten Segments	33.1	55.3	69.3	68.7

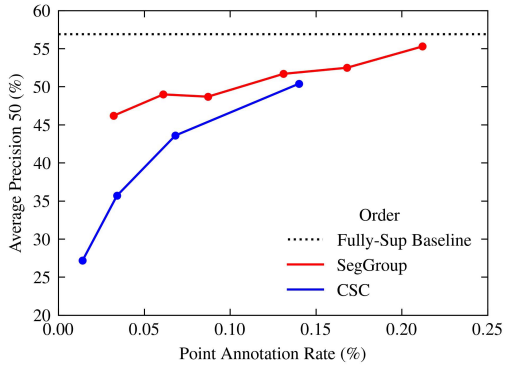


Fig. 6. Point cloud instance segmentation AP 50 results (%) of PointGroup [20] network trained by SegGroup and CSC [25] with different point annotation rates on the ScanNet validation set.

of Top-3 are better than Top-1 on semantic segmentation, while worse on instance segmentation. Top-3 annotations can increase variety in seg-level labels across instances, while lower the labeling quality of each instance. The results show that semantic segmentation supervision benefits more from labeling variety across instances, while instance segmentation is more sensitive to instance-specific labeling quality.

When the annotation number increases, we find that we can achieve better performance in Table IX. We increased the annotation number by allowing more than one annotation for each instance. For fairness, we only compared results of mechanical seg-level labels generated from the largest N segments for every instance. Following the settings in Table VIII, we also employed PointGroup [20] and KPConv [14] to train a point cloud instance segmentation model and a point cloud semantic segmentation model. Both instance and semantic

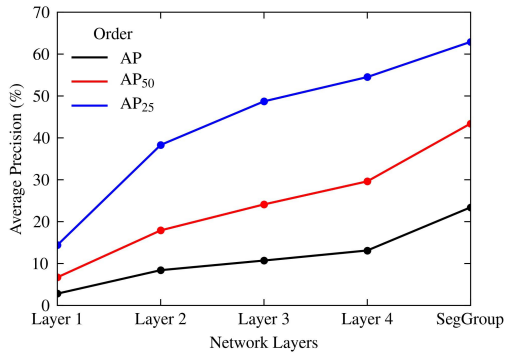


Fig. 7. Point cloud instance segmentation results (%) of PointGroup network trained with pseudo labels from different layers of SegGroup on the ScanNet validation set. Layer names in X-axis are the same as the layer names in Table II.

segmentation results show that the performance can be further boosted to be closer to the fully supervised baseline by the addition of annotation numbers per instance. When annotating ten segments per instance, we can even reach the same semantic segmentation performance with the fully supervised baseline. In Figure 6, we also compare the AP 50 results of our SegGroup in Table IX with the AP 50 results of CSC [25] on instance segmentation. Experimental results validate the effectiveness of our method on the overlapping portion of the point annotation rate.

We also compared the point cloud instance segmentation performance using the generated pseudo labels from different layers of SegGroup. Fig. 7 shows the results of PointGroup [20] network trained with pseudo labels from different layers of SegGroup network on the ScanNet validation set. Layer names in Fig. 7 are the same as the layer names in Table II. As the number of labeled segments increases during pseudo label generation in Table II, the segmentation performance in Fig. 7 becomes better.

V. Conclusion

In this paper, we have exploited the importance of instance locations for 3D scene segmentation and designed a weakly-supervised point cloud segmentation task for full exploitation. More specifically, we click on one point per instance to indicate its location and extend these location annotations into segments as seg-level labels by over-segmentation. We further design a segment grouping network (SegGroup) to generate point-level pseudo labels by grouping seg-level annotated segments into instances hierarchically, so that existing strong-supervised methods can directly consume the pseudo labels for training. Experimental results on both instance and semantic segmentation show that SegGroup effectively generates high-quality point-level pseudo labels from the locations of instances given the seg-level labels, which well balances the annotation cost and segmentation accuracy.

References

[1] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 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. 5105–5114.

[3] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or, "MeshCNN: A network with an edge," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–12, 2019.

[4] J. Schult, F. Engelmann, T. Kontogianni, and B. Leibe, "DualConvMesh-Net: Joint geodesic and Euclidean convolutions on 3D meshes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8612–8622.

[5] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5648–5656.

[6] Z. Wu *et al.*, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1912–1920.

[7] A. Dai and M. Nießner, "3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 452–468.

[8] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 945–953.

[9] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 165–174.

[10] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3D reconstruction in function space," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4460–4470.

[11] Y. Duan, H. Zhu, H. Wang, L. Yi, R. Nevatia, and L. J. Guibas, "Curriculum DeepSDF," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 51–67.

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

[13] 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.

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

[15] S. Cheng, X. Chen, X. He, Z. Liu, and X. Bai, "PRA-Net: Point relation-aware network for 3D point cloud analysis," *IEEE Trans. Image Process.*, vol. 30, pp. 4436–4448, 2021.

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

[17] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3947–3956.

[18] J. Hou, A. Dai, and M. Nießner, "3D-SIS: 3D semantic instance segmentation of RGB-D scans," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4421–4430.

[19] B. Yang *et al.*, "Learning object bounding boxes for 3D instance segmentation on point clouds," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1–10.

[20] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "PointGroup: Dual-set point grouping for 3D instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4867–4876.

[21] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal ConvNets: Minkowski convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3075–3084.

[22] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5828–5839.

[23] J. Wei, G. Lin, K.-H. Yap, T.-Y. Hung, and L. Xie, "Multi-path region mining for weakly supervised 3D semantic segmentation on point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 4384–4393.

[24] X. Xu and G. H. Lee, "Weakly supervised semantic point cloud segmentation: Towards 10× fewer labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13706–13715.

[25] J. Hou, B. Graham, M. Nießner, and S. Xie, "Exploring data-efficient 3D scene understanding with contrastive scene contexts," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15587–15597.

[26] Q. Hu *et al.*, "SQN: Weakly-supervised semantic segmentation of large-scale 3D point clouds with 1000x fewer labels," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 1–14.

[27] Y. Zhang, Y. Qu, Y. Xie, Z. Li, S. Zheng, and C. Li, "Perturbed self-distillation: Weakly supervised large-scale point cloud semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15520–15528.

[28] Z. Liu, X. Qi, and C.-W. Fu, "One thing one click: A self-training approach for weakly supervised 3D semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 1726–1736.

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

[30] H. Shuai, X. Xu, and Q. Liu, "Backward attentive fusing network with local aggregation classifier for 3D point cloud semantic segmentation," *IEEE Trans. Image Process.*, vol. 30, pp. 4973–4984, 2021.

[31] D. Li, G. Shi, Y. Wu, Y. Yang, and M. Zhao, "Multi-scale neighborhood feature extraction and aggregation for point cloud segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 6, pp. 2175–2191, Jun. 2021.

[32] H. Liu, Y. Guo, Y. Ma, Y. Lei, and G. Wen, "Semantic context encoding for accurate 3D point cloud segmentation," *IEEE Trans. Multimedia*, vol. 23, pp. 2045–2055, 2020.

[33] 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. (ICCV)*, Oct. 2019, pp. 1607–1616.

[34] Q. Hu *et al.*, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11108–11117.

[35] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4558–4567.

[36] S. Xie, S. Liu, Z. Chen, and Z. Tu, "Attentional ShapeContextNet for point cloud recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4606–4615.

[37] J. Yang *et al.*, "Modeling point clouds with self-attention and Gumbel subset sampling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3323–3332.

[38] L. Han, T. Zheng, L. Xu, and L. Fang, "OccuSeg: Occupancy-aware 3D instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2940–2949.

[39] P. O. Pinheiro and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1713–1721.

[40] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1742–1750.

[41] L. Jing, Y. Chen, and Y. Tian, "Coarse-to-fine semantic segmentation from image-level labels," *IEEE Trans. Image Process.*, vol. 29, pp. 225–236, 2020.

[42] C. Redondo-Cabrera, M. Baptista-Rios, and R. J. López-Sastre, "Learning to exploit the prior network knowledge for weakly supervised semantic segmentation," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3649–3661, Jul. 2019.

[43] Z. Shi, Y. Yang, T. M. Hospedales, and T. Xiang, "Weakly-supervised image annotation and segmentation with objects and attributes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2525–2538, Dec. 2017.

[44] J. Dai, K. He, and J. Sun, "BoxSup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1635–1643.

[45] C.-C. Hsu, K.-J. Hsu, C.-C. Tsai, Y.-Y. Lin, and Y.-Y. Chuang, "Weakly supervised instance segmentation using the bounding box tightness prior," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 6586–6597.

[46] A. Khoreva, R. Benenson, J. Hosang, M. Hein, and B. Schiele, "Simple does it: Weakly supervised instance and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 876–885.

[47] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2921–2929.

[48] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan, "Object region mining with adversarial erasing: A simple classification to semantic segmentation approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1568–1576.

[49] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder–decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 801–818.

[50] C. Rother, V. Kolmogorov, and A. Blake, "'GrabCut' interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.

[51] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with Gaussian edge potentials," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 109–117.

[52] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei, "What's the point: Semantic segmentation with point supervision," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 549–565.

[53] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "ScribbleSup: Scribble-supervised convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3159–3167.

[54] X. Li, H. Ma, and X. Luo, "Weaklier supervised semantic segmentation with only one image level annotation per category," *IEEE Trans. Image Process.*, vol. 29, pp. 128–141, 2020.

[55] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, Sep. 2004.

[56] A. Karpathy, S. Miller, and L. Fei-Fei, "Object discovery in 3D scenes via shape analysis," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 2088–2095.

[57] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–10.

[58] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. Int. Conf. Comput. Statist.*, 2010, pp. 177–186.

[59] S. Chen, J. Fang, Q. Zhang, W. Liu, and X. Wang, "Hierarchical aggregation for 3D instance segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15467–15476.

[60] Z. Liang, Z. Li, S. Xu, M. Tan, and K. Jia, "Instance segmentation in 3D scenes using semantic superpoint tree networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 2783–2792.

[61] B. Zhang and P. Wonka, "Point cloud instance segmentation using probabilistic embeddings," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8883–8892.

[62] T. He, C. Shen, and A. van den Hengel, "DyCo3D: Robust instance segmentation of 3D point clouds through dynamic convolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 354–363.

[63] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, and M. Nießner, "3D-MPA: Multi-proposal aggregation for 3D semantic instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9031–9040.

[64] J. Lahoud, B. Ghanem, M. R. Oswald, and M. Pollefeys, "3D instance segmentation via multi-task metric learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9256–9266.

[65] A. Nekrasov, J. Schult, O. Litany, B. Leibe, and F. Engelmann, "Mix3D: Out-of-context data augmentation for 3D scenes," in *Proc. Int. Conf. 3D Vis. (3DV)*, Dec. 2021, pp. 116–125.

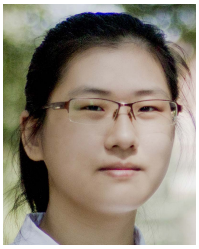
[66] Z. Hu *et al.*, "VMNet: Voxel-mesh network for geodesic-aware 3D semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15488–15498.

[67] A. Kundu *et al.*, "Virtual multi-view fusion for 3D semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 518–535.

[68] Z. Hu, M. Zhen, X. Bai, H. Fu, and C.-L. Tai, "JSENet: Joint semantic segmentation and edge detection network for 3D point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 222–239.

[69] F. Zhang, J. Fang, B. Wah, and P. Torr, "Deep FusionNet for point cloud semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 644–663.

[70] J. Zhang, C. Zhu, L. Zheng, and K. Xu, "Fusion-aware point convolution for online semantic 3D scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4534–4543.
[71] L. Jiang, H. Zhao, S. Liu, X. Shen, C.-W. Fu, and J. Jia, "Hierarchical point-edge interaction network for point cloud semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 10433–10441.
[72] H. Lei, N. Akhtar, and A. Mian, "SegGCN: Efficient 3D point cloud segmentation with fuzzy spherical kernel," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11611–11620.
[73] J. Huang, H. Zhang, L. Yi, T. Funkhouser, M. Nießner, and L. J. Guibas, "TextureNet: Consistent local parametrizations for learning from high-resolution signals on meshes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4440–4449.
[74] 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. 828–838.



An Tao (Graduate Student Member, IEEE) received the B.Eng. degree from the School of Information Science and Engineering, Southeast University, Nanjing, China, in 2019. She is currently pursuing the Ph.D. degree with the Department of Automation, Tsinghua University, Beijing, China. Her current research interest is 3D vision. She has served as a Reviewer for several conferences, e.g., CVPR, ICCV, ECCV, ICME, and 3DV.

Yueqi Duan (Member, IEEE) received the B.S. and Ph.D. degrees from the Department of Automation, Tsinghua University, in 2014 and 2019, respectively. From 2019 to 2021, he worked as a Postdoctoral Researcher with the Computer Science Department, Stanford University. He is currently an Assistant Professor with the Department of Electronic Engineering, Tsinghua University. He has published more than ten scientific articles in the top journals and conferences, including TPAMI, TIP, CVPR, and ECCV. His research interests include computer vision and pattern recognition. He serves as the Area Chair of ICME from 2020 to 2022, and a Regular Reviewer for a number of journals and conferences, e.g., TPAMI, IJCV, TIP, CVPR, ICCV, ECCV, ICML, NeurIPS, and SIGGRAPH. He was awarded the Excellent Doctoral Dissertation of Chinese Association for Artificial Intelligence (CAAI) in 2020.

Yi Wei received the B.S. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2019, where he is currently pursuing the Ph.D. degree with the Department of Automation. He has authored six scientific articles in this area, including CVPR, ECCV, and ICRA. His research interests lie in 3D vision, computer graphics, and robotics, especially focusing on 3D scene understanding and 3D reconstruction. He serves as a Reviewer for a number of journals and conferences, e.g., TIP, TCSVT, CVPR, and ICCV.

Jiwen Lu (Senior Member, IEEE) received the B.Eng. degree in mechanical engineering and the M.Eng. degree in electrical engineering from the Xi'an University of Technology, Xi'an, China, in 2003 and 2006, respectively, and the Ph.D. degree in electrical engineering from Nanyang Technological University, Singapore, in 2012. He is currently an Associate Professor with the Department of Automation, Tsinghua University, Beijing, China. His current research interests include computer vision and pattern recognition. He was/is a member of the Image, Video and Multidimensional Signal Processing Technical Committee, the Multimedia Signal Processing Technical Committee, the Information Forensics and Security Technical Committee of the IEEE Signal Processing Society, the Multimedia Systems and Applications Technical Committee, and the Visual Signal Processing and Communications Technical Committee of the IEEE Circuits and Systems Society. He is a fellow of IAPR. He was a recipient of the National Natural Science Funds for Distinguished Young Scholar. He serves as the General Co-Chair for the International Conference on Multimedia and Expo (ICME) 2022 and the Program Co-Chair for the ICME 2020, the International Conference on Automatic Face and Gesture Recognition (FG) 2023, and the International Conference on Visual Communication and Image Processing (VCIP) 2022. He serves as the Co-Editor-of-Chief for *Pattern Recognition Letters* and an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and the IEEE TRANSACTIONS ON BIOMETRICS, BEHAVIOR, AND IDENTITY SCIENCES.

Jie Zhou (Senior Member, IEEE) received the B.S. and M.S. degrees from the Department of Mathematics, Nankai University, Tianjin, China, in 1990 and 1992, respectively, and the Ph.D. degree from the Institute of Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology (HUST), Wuhan, China, in 1995. From 1995 to 1997, he worked as a Postdoctoral Fellow with the Department of Automation, Tsinghua University, Beijing, China, where he has been a Full Professor with the Department of Automation since 2003. In recent years, he has authored more than 100 papers in peer-reviewed journals and conferences. Among them, more than 40 papers have been published in top journals and conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON IMAGE PROCESSING, and CVPR. His research interests include computer vision, pattern recognition, and image processing. He is an IAPR Fellow. He received the National Outstanding Youth Foundation of China Award. He is an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and two other journals.