One Thing One Click: A Self-Training Approach for Weakly Supervised 3D Semantic Segmentation

Zhengzhe Liu\(^1\)  \hspace{1em}  Xiaojuan Qi\(^2\)  \hspace{1em}  Chi-Wing Fu\(^1\)  \\
\(^1\)The Chinese University of Hong Kong  \hspace{1em}  \(^2\)The University of Hong Kong  \\
\{zzliu,cwfu@cse.cuhk.edu.hk  xjqi@eee.hku.edu.hk

Abstract

Point cloud semantic segmentation often requires large-scale annotated training data, but clearly, point-wise labels are too tedious to prepare. While some recent methods propose to train a 3D network with small percentages of point labels, we take the approach to an extreme and propose “One Thing One Click,” meaning that the annotator only needs to label one point per object. To leverage these extremely sparse labels in network training, we design a novel self-training approach, in which we iteratively conduct the training and label propagation, facilitated by a graph propagation module. Also, we adopt a relation network to generate the per-category prototype and explicitly model the similarity among graph nodes to generate pseudo labels to guide the iterative training. Experimental results on both ScanNet-v2 and S3DIS show that our self-training approach, with extremely-sparse annotations, outperforms all existing weakly supervised methods for 3D semantic segmentation by a large margin, and our results are also comparable to those of the fully supervised counterparts.

1. Introduction

The success of 3D semantic segmentation benefits a lot from the large annotated training data. However, annotating a large amount of point cloud data is exhausting and costly. Taking ScanNet-v2\(^7\) as an example, it takes 22.3 minutes to annotate one scene on average. It is a great burden to annotate the whole data set, which includes 1,513 scenes, thus potentially restricting further applications that require larger scale data. Thus, efficient approaches to facilitate 3D data annotation are highly desirable.

Very recently, some methods [47, 46, 50] were proposed to reduce efforts to annotating 3D point clouds. Though they improve annotation efficiency, various issues remain. Scene-level annotation in [47] could impose negative effects on the model in the absence of localization information, whereas sub-cloud annotation in [47] requires an extra burden to first divide the input into subclouds and then repeatedly annotate semantic categories in individual subclouds. The 2D image annotation approach [46] requires extra labor to prepare a 2D image annotation, which is also a tedious task on its own. Xu et al. [50] presume that the labeled points follow a uniform distribution. Such a requirement can be achieved by...
subsampling from a fully-annotated dataset, but is hard for
the annotators to follow in practice.

In this work, we also aim to reduce the amount of neces-
sary annotations on point clouds, but we take the approach
to an extreme by proposing “One Thing One Click,” so the
annotator only needs to label one single point per object.
To further relieve the annotation burden, such a point can
be randomly chosen, not necessarily at the center of the ob-
ject. On average, it takes less than 2 minutes to annotate a
ScanNet-v2 scene with our “One Thing One Click” scheme
(see an example annotation in Figure 2 (b), which contains
only 13 clicks), which is more than 10x faster compared
with the original ScanNet-v2 annotation scheme.

However, directly training a network on the extremely-
sparse labels from our annotating scheme (less than 0.02%
in ScanNet-v2 and S3DIS) will easily make the network
overfit the limited data and restrict its generalization ability.
Hence, it raises a question that “can we achieve a perform-
ance comparable with a fully supervised baseline given the
extremely-sparse annotations?” To meet such a challenge,
we design a self-training approach with a label-propagation
mechanism for weakly supervised semantic segmentation.
On the one hand, with the prediction result of the model, the
pseudo labels can be expanded to unknown regions through
our graph propagation module. On the other hand, with
richer and higher quality labels being generated, the model
performance can be further improved. Thus, we conduct the
label propagation and network training iteratively, forming a
closed loop to boost the performance of each other.

A core problem of label propagation is how to measure
the similarity among nodes. Previous works [54, 5, 52] build
a graph model upon 2D pixels and measure the similarity
with low-level image features, e.g., coordinates and colors.
In contrast, our graph is built upon the 3D super-voxels with
more complex geometric structures and a variable number of
points in each group. Hence, existing hand-craft features can-
not fully reveal the similarity among nodes in our case. To
resolve this problem, we further propose a relation network
to leverage 3D geometrical information for similarity learn-
ing among the graph nodes in 3D. The geometrical similarity
and learned similarity are integrated together to facilitate
label propagation. To effectively train the relation network
with the extremely-sparse and category-unbalanced data, we
further propose to generate a category-wise prototype with a
memory bank for better similarity measurement.

Experiments conducted on two public data sets ScanNet-
v2 and S3DIS manifest the effectiveness of the proposed
method. With just around 0.02% point annotations, our ap-
proach surpasses all existing weakly supervised approaches
(which employ far more labels) for 3D point cloud segmen-
tation by a large margin, and our approach even achieves
results that are comparable with a fully supervised counter-
part; see Figure 1. These results manifest the high efficiency
of our “One Thing One Click” scheme for 3D point cloud an-
notation and the effectiveness of our self-training approach
for weakly supervised 3D semantic segmentation.

2. Related Work

Semantic Segmentation for Point Cloud Approaches for
3D semantic segmentation can be roughly divided into point-
based methods and voxel-based methods. Point-based net-
works take raw point clouds as input. Along this line of
works, PointNet [33] and PointNet++ [34] are the pioneering
ones. Afterward, convolution-based methods [23, 44, 48, 4]
were also proposed for 3D semantic segmentation on point
clouds. Recently, Kundu et al. [19] proposed to fuse features
from multiple 2D views for 3D semantic segmentation. To
aggregate together the geometrically-homogeneous points,
Landrieu et al. [21] modeled a point cloud as a super point
graph. Inspired by [21], we expand the sparse labels to ge-
ometrically homogeneous super-voxels to generate initial
pseudo labels for the first-iteration network training.

Voxel-based networks take the regular voxel-grids as input
instead of the raw data [43, 37, 11, 40, 8]. The recently-
proposed methods SparseConv [12], MinkowskiNet et al. [6],
and OccuSeg et al. [14] are among the representative works
in this branch. In this paper, we adopt the 3D-UNet architec-
ture described in [12] as the backbone architecture due to its
high performance and applicability.

Weakly Supervised 3D Semantic Segmentation Compared with fully supervised 3D semantic segmentation,
weakly supervised 3D semantic segmentation is relatively
under-explored. After early works [28, 13] in this area, very
recently, Wei et al. [47] utilized the Class Activation Map to
generate pseudo point-wise labels from sub-cloud-level an-
notations. The performance is, however, limited by the lack
of localization information. Wang et al. [46] back-projected
2D image annotations to 3D space to produce labels in point
clouds. However, annotating large-scale semantic segmen-
tation on 2D images is also laborious. Also, the visibility
prediction branch adds to the complexity of the network.
Xu et al. [50] achieve a performance close to fully sup-
ervised with less than 10% labels. However, they require the
annotations to be uniformly-distributed in the point cloud,
which is practically very hard for the annotators to follow.

Different from the existing works, we propose a new
self-training approach with a graph propagation module,
in which the network training and label propagation are
conducted iteratively. Our approach largely reduces the
reliance on the quality of the initial annotation and achieves
top performances, compared with existing weakly supervised
methods, while using only extremely-sparse annotations.

Self-Training Self-training for weakly supervised 2D im-
age understanding has been intensively explored. To reduce

1727
the annotation burden for 2D images, researchers proposed a variety of annotation approaches, e.g., image-level categories [35, 29, 55, 1], points [3, 22], scribbles [24, 45, 53], bounding boxes [9], etc. With the weak supervision, a self-training approach can learn to expand the limited annotations to unknown regions in the domain. As far as we know, this is the first work that explores self-training for weakly supervised 3D semantic segmentation.

3. Methodology

3.1. Overview

With “One Thing One Click,” we only need to annotate a point cloud with one point per object, as Figure 3 (c) shows, and these points can be chosen at random to alleviate the annotation burden. Procedure-wise, given such sparse annotations, we first over-segment the point cloud $X = \{p_i\}$ into geometrically homogeneous super-voxels $V = \{v_j\}$, where $\cup_j v_j = X$ and $v_j \cap v_{j'} = \emptyset$ for $v_j \neq v_{j'}$. Note that throughout the paper, we use $i$ and $j$ as the indices for points and super-voxels, respectively. Based on the super-voxel partition, we can produce initial pseudo labels of the point cloud by spreading each label to all the points locally in the super-voxel that contains the annotated point. However, as Figure 3 (d) shows, the labels are still very sparse. More importantly, the propagated labels distribute mainly around the initially-annotated points, which are far from the ideal uniform distribution for weakly semantic segmentation, as employed in [50].

An important insight in our approach is to iteratively propagate the sparse annotations to unknown regions in the point cloud, while training the network model to guide the propagation process. To achieve this, we adopt the 3D semantic segmentation network $\Theta$ (the blue regions in Figure 3) to learn to propagate via a graph model (Figure 3 (f)). Further, we design the relation network $R$ (the orange regions in Figure 3) to explicitly model the feature similarity among the graph nodes. Afterward, predictions with high confidence are further employed as the updated pseudo labels for training the network in the next iteration (Figure 3 (g)). This iterative self-training approach couples the label propagation and network training, enabling us to significantly enhance the segmentation quality, as revealed earlier in Figure 1.

In this section, we first present our 3D semantic segmentation network for point-wise semantic prediction (Section 3.2), then our label propagation mechanism with a graph model and the relation network for similarity learning (Section 3.3). Afterward, we describe the self-training approach that evolves the above modules alternatively (Section 3.4).

3.2. 3D Semantic Segmentation Network

We adopt the 3D U-Net architecture [12] as the backbone, denoted as $\Theta$. Its input is point cloud $X$ of $N$ points (Figure 3 (a)). Each point has 3D coordinates $p_i$ and color $c_i$, where $i \in \{1, ..., N\}$. The network predicts the probability of each semantic category $P(y_i, \bar{c}|p_i, c_i, \Theta)$ of each point $p_i$, where $\bar{c}$ is the ground truth category of point $p_i$. The network is trained with the softmax cross-entropy loss below:

$$L_s = \frac{1}{N} \sum_{i=1}^{N} - \log P(y_i, \bar{c}|p_i, c_i, \Theta).$$

(1)

In the first iteration, the network is trained with the initial pseudo labels, as shown in Figure 3 (d). In subsequent
iterations, the network is trained with the updated pseudo labels, as shown in Figure 3 (g), which will be detailed below.

3.3. Pseudo Label Generation by Graph Propagation

To facilitate the network training, we propose a graph propagation mechanism to effectively propagate labels to unknown regions. We also propose the relation network to explicitly learn the similarity among the super-voxels to facilitate the label propagation process and complement 3D U-Net.

Graph Construction To start, we leverage the 3D geometrically homogeneous super-voxels to build a graph. Compared with building on points, our graph has significant fewer nodes to facilitate efficient label propagation.

To derive the prediction $P(y_j, v_j, \Theta)$ of the $j$-th super-voxel, we apply a super-voxel pooling to aggregate the semantic prediction of the $n_j$ points in $v_j$ as below:

$$P(y_j, v_j, \Theta) = \frac{1}{n_j} \sum_{i} P(y_i, p_i, c_i, \Theta), \text{ where } p_i \in v_j,$$

(2)

where $P(y_i, p_i, c_i, \Theta)$ is the probability of $p_i$ in class $c$.

To build the graph, we treat each super-voxel as a graph node and compute the similarity between each pair of super-voxels $v_j, v_{j'}$, which is represented as an edge. Further, to propagate labels to unknown regions through the graph, we formulate it as an optimization problem that considers both the network prediction and similarities among the super-voxels to achieve the global optimum with the energy function below similar to Conditional Random Field (CRF).

$$E(Y|V) = \sum_{j} \psi_u(y_j|V, \Theta) + \sum_{j<j'} \psi_p(y_j, y_{j'}|V, \mathcal{R}, \Theta)$$

(3)

where $\mathcal{R}$ is the relation network to be detailed later. The unary term $\psi_u(y_j|V, \Theta)$ represents the super-voxel pooled prediction of the 3D U-Net $P(y_j)$ on super-voxel $v_j$. Specifically, it denotes the minus log probability of predicting super-voxel $v_j$ to have label $y_j$. We define it as below:

$$\psi_u(y_j|V, \Theta) = -\log P(y_j|V, \Theta)$$

(4)

The pairwise term $\psi_p(j_k)$ in Equation 3 represents the similarity between super-voxels $v_j$ and $v_{j'}$. We employ both the low-level features and learned features for measuring the similarity, as shown in Equation 5 below:

$$\psi_p(y_j, y_{j'}|V) = \mathbb{I} (y_j, y_{j'}) \exp \left\{ -\lambda_p \frac{||c_j - c_{j'}||^2}{2\sigma_c^2} - \lambda_p \frac{||p_j - p_{j'}||^2}{2\sigma_p^2} - \lambda_u \frac{||u_j - u_{j'}||^2}{2\sigma_u^2} - \lambda_f \frac{||f_j - f_{j'}||^2}{2\sigma_f^2} \right\}$$

(5)

where $\mathbb{I} (y_j, y_{j'})$ is 1, if $v_j$ and $v_{j'}$ have different predicted labels, and 0 otherwise. The pairwise term means that the cost will be higher if super-voxels with similar features are predicted to be different classes. Here, $c_j, c_{j'}, p_j, p_{j'}, u_j, u_{j'}, f_j, f_{j'}$ are the normalized mean color, mean coordinates and mean 3D U-Net feature, respectively, of super-voxels $v_j$ and $v_{j'}$. Unlike existing works [34, 5, 52], which build the graph on 2D image pixels, we build our graph on 3D super-voxels, which have irregular and complex geometrical structures, as shown in the supplementary material. Therefore, hand-crafted features $p_j, p_{j'}$ and $c_j, c_{j'}$ have inferior capability to measure the similarity between super-voxels. To address this issue, we propose the Relation Network to better leverage the 3D geometrical information and explicitly learn the similarity among super-voxels.

Relation Network Existing works Co-Training [36] and Tri-net [10] showed that semi-supervised training benefits from having two complementary tasks or components. In our framework, we propose a relation net to complement the 3D U-Net. The relation network $\mathcal{R}$ shares the same backbone architecture as the 3D U-Net $\Theta$ except for removing the last category-wise prediction layer. It aims to predict a category-related embedding $f_j$ for each super-voxel $v_j$ as the similarity measurement. Similar to Equation 2, $f_j$ is the per super-voxel pooled feature in $\mathcal{R}$. In other words, the relation network groups the embeddings of same category together, while pushing those of different categories apart. To this end, we propose to learn a prototypical embedding for each category, inspired by the Prototypical Network [39].

However, the per-category prototypes in [39] are fully determined by the sampled mini-batch, and may deviate from the actual categorical center. Consequently, they may not be stable and could keep changing during the training, thereby hard to converge. To assist the training of the relation network with sparse and unbalanced training data, we present a memory bank $K = \{k\}$ to generate one categorical prototype for each category, instead of simply regarding the average embedding as the prototype as in [39].

The embedding $f_j$ generated by $\mathcal{R}$ serves as a “query,” and we compare it with the corresponding “key” $k_c$ in the memory bank with a dot product. The two modules are optimized simultaneously with contrastive learning [30] as below.

$$L_c = \frac{1}{M} \sum_j (\log \frac{f_j \cdot k_c/\tau}{\sum_c f_j \cdot k_c/\tau}),$$

(6)

where $\tau$ is a temperature hyperparameter [49] and $c$ is the ground truth category of $v_j$. The contrastive learning is equivalent to a c-way softmax classification task.

Following [15], we update the key representations via a moving average with momentum as shown below

$$k_{c} \leftarrow mk_{c} + (1 - m)f_j,$$

(7)
3.4. Self-Training

With the energy function in Equation 3, we propose a self-training approach to update networks \( \Theta \) and \( \mathcal{R} \), and also the pseudo labels \( Y \) iteratively, as Algorithm 1 outlines. The self-training is started by the “One Thing One Click” annotations and the pre-constructed super-voxel graph. In each iteration, we fix network parameters \( \Theta, \mathcal{R} \) and update label \( Y \), and vice versa. There are two steps in each iteration.

- With \( \Theta \) and \( \mathcal{R} \) fixed, the label propagation is conducted to minimize the energy function in Equation 3. Then, the predictions with high confidence are updated as the added pseudo labels for training the two networks in the next iteration. The confidence of super-voxel \( v_j \), denoted as \( C_j \), is the average of the minus \( \log \) probability of all \( n_j \) points in \( v_j \) after the label propagation:

\[
C_j = -\frac{1}{n_j} \sum_{i} \log P(y_i|p_i, V, \Theta, \mathcal{R}, G), \quad \text{where } p_i \in v_j,
\]

where \( G \) denotes the graph propagation. (8)

- With pseudo labels \( Y, \Theta \) and \( \mathcal{R} \) are optimized with softmax loss and contrastive loss, respectively.

4. Experiments

Datasets Our experiments are conducted on two large 3D semantic segmentation datasets – ScanNet-v2 [7] and S3DIS [2]. ScanNet-v2 [7] contains 1513 3D scans of 20 semantic categories. We annotate the official training set with our “One Thing One Click” scheme, and evaluate on the original validation and test set. S3DIS [2] contains 3D scans of 271 rooms containing 13 categories. We follow the official train/validation split to annotate on Area 1,2,3,4,6 and report the performance on Area 5.

“One Thing One Click” Annotation Details In order to ensure the randomness of point selection in annotation, we simulate the annotation procedure by selecting a single point inside an object with the same probability for the following experiments. In ScanNet-v2, only 19.74 points per scene are annotated on average with “One Thing One Click” scheme, while this number in the original ScanNet-v2 is 108875.9. In S3DIS, only 36.15 points in each room are annotated on average using “One Thing One Click”, while the original S3DIS has 193797.1 points annotated in each room.

Implementation Details We implement all the modules of our self-training framework including the mean-field solver [18] for label propagation with the PyTorch [32] framework based on the implementation of [17]. Following [17], due to the GPU capacity, we randomly choose 250k points if the scene contains more points in training. In inference, the network takes the whole scene as input. We use the mesh segment results [7] as super-voxels for ScanNet-v2, and the geometrical partition results described in [21] for S3DIS super-voxel partition. We set the hyper-parameters \( D = 32, T = 0.9, s = 20, \tau = 0.07, m = 0.9, \sigma_c = \sigma_p = \sigma_u = \sigma_f = 1, \lambda_c = \lambda_p = \lambda_u = \lambda_f = 1 \) with a small validation set. We found that the self-training converges after five iterations. After that, more iterations training only brings very minor improvements.

4.1. Evaluations on ScanNet-v2

Comparing with Existing Methods Table 1 reports the benchmark result on ScanNet-v2 test set. The baselines can be roughly divided into two branches. (i) Fully supervised approaches with 100% supervision, including several
representative works in 3D semantic segmentation. These methods are the upper bounds of weakly supervised ones. (ii) Weakly supervised approaches, including a recent work [47].

With less than 0.02% annotated points, our result (69.1% mIoU) outperforms many existing works with full supervision. As for weakly supervised approaches, MPRM [47] is trained with scene-level or subcloud-level labels. The scene-level annotation leads to an inferior performance of 24.4%, and the subcloud-level annotation takes around 3 minutes per scene as reported in [47], which is longer than our “One Thing One Click” scheme (2 minutes). More importantly, our result outperforms [47] by more than 26% mIoU.

**Comparing with Our Baselines** In this section, we first present three important baselines as shown in Table 2 on ScanNet-v2 validation set.

- Table 2 “Our fully sup baseline” is trained with the official 100% annotation provided by ScanNet-v2. It serves as the upper bound of our method.
- The model directly trained with the raw annotated points as Figure 3 (c) cannot converge well due to the extreme sparsity of the training data.
- Table 2 “One Thing One Click††”. The model trained with the initial pseudo labels as Figure 3 (d) achieves 62.18% mIoU. It serves as the starting point of our self-training approach and is denoted as “our baseline” in the following.

Table 2 “One Thing One Click” manifests that our self-training approach surpasses the baseline by nearly 10% mIoU, attaining a 16% relative improvement. Compared with the fully supervised baseline with the same network architecture, our performance is only 2% lower.

Table 2 “One Thing One Click††” refers to disabling the graph propagation and relation network in inference. Note that they are still being used in training for generating the pseudo labels. This brings no extra computational burden during the inference, but helps to improve nearly 7% mIoU, comparing with the baseline (68.96% vs 62.18%).

The quantitative results in Figure 5 indicate our result (c) is very similar to the fully supervised baseline (e) [12] in ScanNet-v2. Check error maps (d) (f) for better comparison.

**Results on ScanNet-v2 Data-Efficient Benchmark** In this section, we show results on ScanNet-v2 “3D Semantic label with Limited Annotations” benchmark. We report the results on the most challenging setting with only 20 points annotated each scene in Table 1 “Ours on Data Efficient”. We use the fully-supervised baseline with the initial pseudo labels shown in Figure 3 (d). † means disabling graph propagation and relation network during inference, but note that they are still used in training.

Table 2 “Data Efficient †” means disabling the graph propagation and relation network in inference. Note that they are still being used in training for generating the pseudo labels. This brings no extra computational burden during the inference, but helps to improve nearly 7% mIoU, comparing with the baseline (68.96% vs 62.18%).

The quantitative results in Figure 5 indicate our result (c) is very similar to the fully supervised baseline (e) [12] in ScanNet-v2. Check error maps (d) (f) for better comparison.

**Results on ScanNet-v2 Data-Efficient Benchmark** In this section, we show results on ScanNet-v2 “3D Semantic label with Limited Annotations” benchmark. We report the results on the most challenging setting with only 20 points annotated each scene in Table 1 “Ours on Data Efficient”. We use the fully-supervised baseline with the initial pseudo labels shown in Figure 3 (d). † means disabling graph propagation and relation network during inference, but note that they are still used in training for generating the pseudo labels. This brings no extra computational burden during the inference, but helps to improve nearly 7% mIoU, comparing with the baseline (68.96% vs 62.18%).

The quantitative results in Figure 5 indicate our result (c) is very similar to the fully supervised baseline (e) [12] in ScanNet-v2. Check error maps (d) (f) for better comparison.

**Ablation Studies** To further study the effectiveness of self-training, graph propagation and relation network, we conduct ablation studies on these three modules on ScanNet-v2 validation set as shown in Table 3 with single view evaluation.

“3D U-Net” indicates that the labels are propagated only based on the confidence score of the 3D U-Net itself, i.e., the unary term in Equation 3. This ablation is designed to manifest the effectiveness of self-training. The “3D U-Net” column in Table 3 manifests that the performance is consistently improved with self-training strategy even without pairwise energy term in Equation 3 and super-voxel partition.

“3D U-Net+GP” refers to the label propagation with
Figure 4. Pseudo labels for each iteration on ScanNet-v2 training set.

Figure 5. Quantitative results of our method and fully supervised baseline. (d) is the error map of our prediction (c), and (f) is the error map of our fully supervised baseline [12] (e). Red regions indicate the wrong prediction.

Figure 6. The t-SNE visualization of super-voxel features. Different colors and marks (point and plus) indicate different categories. The samples of the same category are better grouped together with our relation network (c), compared with hand-crafted features (a & b).

<table>
<thead>
<tr>
<th>Method</th>
<th>3D U-Net</th>
<th>3D U-Net+GP</th>
<th>3D U-Net+Rel+GP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iter1</td>
<td>60.14</td>
<td>63.83</td>
<td>63.92</td>
</tr>
<tr>
<td>Iter2</td>
<td>62.39</td>
<td>64.74</td>
<td>66.97</td>
</tr>
<tr>
<td>Iter3</td>
<td>64.83</td>
<td>66.10</td>
<td>68.40</td>
</tr>
<tr>
<td>Iter4</td>
<td>65.81</td>
<td>67.78</td>
<td>70.01</td>
</tr>
<tr>
<td>Iter5</td>
<td>65.91</td>
<td>67.92</td>
<td>70.45</td>
</tr>
</tbody>
</table>

Table 3. Ablation studies. “GP” indicates the graph propagation, and “Rel” means the relation network. “3D U-Net” refers to propagating labels only with the network prediction itself. “3D U-Net+GP” indicates label propagation with hand-crafted features. “3D U-Net+Rel+GP” indicates label propagation with our relation network. Evaluated on ScanNet-v2 val. set with single view testing.

As shown in Figure 4, the generated pseudo labels for each iteration expands to unknown regions step by step and finally gets close to the ground truth.

Analysis of Relation Network Further, we study whether the learned embeddings of the relation network outperform the hand-crafted features for similarity measurement. We randomly sample 200 super-voxels for each category in ScanNet-v2, and conduct a t-SNE visualization [26] on them. Figure 6 indicates that the relation network better groups the intra-class embeddings and distinguish the inter-class embeddings compared with hand-crafted features.

4.2. Evaluations on S3DIS

We also evaluate our annotation and training approach on the S3DIS dataset. Only less than 0.02% points in the dataset are annotated with our “One Thing One Click” scheme. To study the performance can be further boosted with
Comparing with Existing Works We also compare with fully supervised approaches and weakly supervised approaches on S3DIS. The latter includes existing works [20, 41] and recent works [50, 46].

As shown in Table 4, with the “One Thing One Click” scheme where less than 0.02% points are annotated, we achieve 50.1% mIoU. With “One Thing Three Clicks” scheme, our performance can be further improved to 55.3% mIoU. The above two results outperform [50] by 5.6% and 10.8% mIoU (0.2% annotations in [50]), and 2.1% and 7.3% mIoU (10% annotations in [50]) respectively.

Wang et al. [46] unprojects 2D semantic labels to 3D space for 3D semantic segmentation. To compare with [46], we first compare with the actual number of annotated points regardless of 2D or 3D. For S3DIS, the number of annotated 2D points (70,496 images with 1080×1080 resolution) is 100× more than the officially annotated 3D points (5.27 × 10^8 in total), so both settings of [46] (100% 2D annotations and 16.7% 2D annotations) actually utilize a large quantity of annotations. Even with a large gap of annotation, the results in Table 4 show that our “One Thing Three Clicks” scheme with only 0.06% 3D annotation outperforms [46] with 100% 2D annotations by nearly 3% mIoU.

In addition, our approach achieves comparable results with several fully supervised methods as shown in Table 4.

Comparing with Our Baselines We follow the similar settings in Section 4.1 to show several baselines for S3DIS.

- Table 4 “Our fully-sup baseline”. The model trained with the full supervision of S3DIS achieves 63.7% mIoU. It serves as the upper bound of our approach.
- The model directly trained with only the annotated points in Figure 3 (c) cannot converge well.
- Table 4 “One Thing One Click” and “One Thing Three Clicks”. The model trained with the annotated supervoxels in Figure 3 (d) achieves 43.7% mIoU for “One Thing One Click” and 48.9% mIoU for “One Thing Three Clicks”. They are used as the baselines to calculate the “relative improvement” of our approach, and are denoted as “our baseline” in the following.

As shown in Table 4 “Rel. Imp.” column, we have 14.6% (“One Thing One Click”) and 13.1% (“One Thing Three Clicks”) relative improvement over our baseline, surpassing the relative improvement of [50], which is 1.1% (with 0.2% annotations) and 5% (with 10% annotations) over their own baselines, by a large margin. The significant improvement of “relative improvement over baseline” manifests the effectiveness of the proposed approach.

<table>
<thead>
<tr>
<th>Method</th>
<th>Supervision (%)</th>
<th>mIoU(%)</th>
<th>Rel. Imp. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PointNet [33]</td>
<td>100%</td>
<td>41.1</td>
<td></td>
</tr>
<tr>
<td>SegCloud [43]</td>
<td>100%</td>
<td>48.9</td>
<td></td>
</tr>
<tr>
<td>TangentConv [42]</td>
<td>100%</td>
<td>52.8</td>
<td></td>
</tr>
<tr>
<td>3D RNN [51]</td>
<td>100%</td>
<td>55.4</td>
<td></td>
</tr>
<tr>
<td>PointCNN [23]</td>
<td>100%</td>
<td>57.3</td>
<td></td>
</tr>
<tr>
<td>SuperpointGraph [21]</td>
<td>100%</td>
<td>58.0</td>
<td></td>
</tr>
<tr>
<td>MinkowskiNet32 [6]</td>
<td>100%</td>
<td>65.4</td>
<td></td>
</tr>
<tr>
<td>Virtual MV-Fusion [19]</td>
<td>100%+2D</td>
<td>65.4</td>
<td></td>
</tr>
<tr>
<td>Our fully-sup baseline</td>
<td>100%</td>
<td>63.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 4. Comparison with existing methods and baselines on the S3DIS Area-5. * indicates baseline models, and † refers to disabling graph propagation and relation network during inference. Note that they are still used in training. “Rel. Imp.” indicates the relative improvement over the baseline. “-” indicates there is no meaningful baseline in this case or it is a baseline itself.

To evaluate without any extra computation burden, we further disable the label propagation and relation network in inference as shown in Table 4 “††”. Note that they are still adopted in training. Our model still attains 13.0% (“One Thing One Click”) and 10.6% (“One Thing Three Clicks”) relative improvement over our baseline in this case.

5. Conclusion

We propose the “One Thing One Click” scheme to efficiently annotate point clouds for weakly supervised 3D semantic segmentation, requiring significantly fewer annotations than the previous approaches. To put this scheme into practice, we formulate a self-training approach to make it feasible for the network to learn from such extremely sparse labels. Specifically, we execute the two key modules in our approach iteratively: expand labels through the graph propagation module and train the network using the updated pseudo labels. Further, we adopt a relation network to explicitly learn the feature similarity among graph nodes with complex 3D structures. Experiments on two large 3D datasets ScanNet-v2 and S3DIS manifest that our approach, with only extremely-sparse annotations, outperforms all the existing weakly supervised methods on 3D semantic segmentation by a large margin, and our results are also comparable to those of the fully supervised counterparts.
References


