Weakly Supervised Segmentation on Outdoor 4D point clouds with Temporal Matching and Spatial Graph Propagation

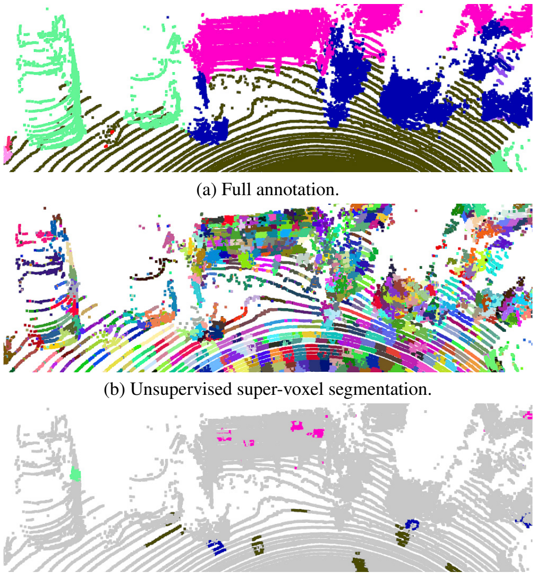
Hanyu Shi[1], Jiacheng Wei[1], Ruibo Li[1,2], Fayao Liu[3] and Guosheng Lin[1,2]*
[1]Nanyang Technological University,
[2]S-Lab for Advanced Intelligence, Nanyang Technological University,
[3] Institute for Infocomm Research, A*STAR,
E-mail: hanyu001@ntu.edu.sg, gslin@ntu.edu.sg

Abstract

Existing point cloud segmentation methods require a large amount of annotated data, especially for the outdoor point cloud scene. Due to the complexity of the outdoor 3D scenes, manual annotations on the outdoor point cloud scene are time-consuming and expensive. In this paper, we study how to achieve scene understanding with limited annotated data. Treating 100 consecutive frames as a sequence, we divide the whole dataset into a series of sequences and annotate only 0.1% points in the first frame of each sequence to reduce the annotation requirements. This leads to a total annotation budget of 0.001%. We propose a novel temporal-spatial framework for effective weakly supervised learning to generate high-quality pseudo labels from these limited annotated data. Specifically, the framework contains two modules: an matching module in temporal dimension to propagate pseudo labels across different frames, and a graph propagation module in spatial dimension to propagate the information of pseudo labels to the entire point clouds in each frame. With only 0.001% annotations for training, experimental results on both SemanticKITTI and SemanticPOSS shows our weakly supervised two-stage framework is comparable to some existing fully supervised methods. We also evaluate our framework with 0.005% initial annotations on SemanticKITTI, and achieve a result close to fully supervised backbone model.

1. Introduction

Recently, outdoor 3D semantic segmentation is attracting more research attention since the introduce of several large datasets, *e.g.*, SemanticKITTI [1] and SemanticPOSS [19]. The outdoor 3D point cloud dataset organises the data as several sequences of point clouds, i.e. 4D point cloud. Then, multiple scans in point cloud sequences are superimposed together and divided as small tiles to reduce manual annotation costs. However, the annotation cost on the small tiles is still high. In SemanticKITTI [1], the annotation on one $100m \times 100m$ tile of highway scene requires an average of 1.5 hours, and the annotation on one tile of more complex scenes requires an average of 4.5 hours. The whole annotation task on SmeanticKITTI requires over 1700 hours. Therefore, the research on accelerating the annotation process is valuable and desirable. We here resort to weakly supervised learning to tackle this annotation issue.


(a) Full annotation.

(b) Unsupervised super-voxel segmentation.

(c) Partially annotated super-voxel level labels.

Figure 1. **An example of super-voxel segmentation and our weak annotation**. Super-voxel segmentation segments the whole point cloud scan into several small units, each containing points within the same class. Therefore, we assign the point level initial annotation to all the points in the same super-voxel.

For indoor 3D point cloud scenes, there are several weakly supervised methods [18, 27, 29, 33] proposed for ac-

[*]Corresponding author: G. Lin. (e-mail: gslin@ntu.edu.sg)

celerating the annotation process. MPRM [29] generates pseudo labels of an indoor 3D scene based on the 2D information. Other approaches [18, 33] annotate a subset of the whole point cloud scene and update weak pseudo labels with the annotated points. For outdoor 3D point cloud scenes, there is no existing weakly supervised segmentation methods available. Directly applying techniques developed for indoor scenes to outdoor scenes can not perform well due to following reasons. Firstly, there is no colour information in outdoor LiDAR point clouds, while methods designed for indoor scenes rely on the colour information to generate and smooth the pseudo labels. Secondly, a typical outdoor point cloud scene contains about 100,000 points for a $150m \times 150m$ area, which is much more sparse than an indoor point cloud scene. Thirdly, as a single outdoor 4D point cloud contains several corresponding point cloud scans, methods proposed for single point cloud scans in the indoor case require extra burdens to generate pseudo labels for each point cloud scan separately.

In this work, we propose a novel weakly supervised framework to reduce the annotation cost in the outdoor point cloud scenario. We exploit the temporal information among point cloud sequences and only annotate 0.1% points in one frame per 100-frame sequence in 4D point clouds. However, training on a weakly labelled dataset with only 0.001% annotated points is unable to learn good features for achieving satisfiable performance. This problem can be concluded as the cold-start problem. To generate more supervision at minimum annotation cost, we apply an efficient super-voxel segmentation [17] on the dataset and assign the labels of annotated points to their belonged super-voxels. Inspired by ScanNet [7] and OTOC [18], super-voxel segmentation segment a point cloud into several small groups, and the points in each group share a same semantic label. We show an example of our annotations in Figure 1.

We then design two modules, *i.e.*, **temporal matching (TM)** and **spatial graph propagation (SGP)**, to propagate the annotations to the whole dataset. TM is designed to generate seeding points in different frames by temporal propagation. For TM , we design two approaches with greedy matching and optimal transport matching. SGP further propagates the searched results to the whole point cloud scene in the spatial dimension.

Furthermore, we propose a two-stage training strategy, which consists of a seed point propagation stage and a dense scene propagation stage. Firstly, the seed point propagation stage propagates initial annotations only along the temporal dimension with TM to generate high-quality pseudo labels under the cold-start scenario. We improve the feature quality by training a new segmentation model on the small amount of high-quality pseudo labels.

In the second stage, we use the new segmentation model from the previous stage to generate features, and based on the new features, we use a dense scene propagation strategy to combine TM and SGP to propagate the label information to the whole dataset. We continue training the model from the previous stage with more pseudo labels to improve the performance further. We evaluate our method on two outdoor segmentation datasets, *i.e.*, SemanticKITTI [1] and SemanticPOSS [19]. Experimental results show that our method achieves comparable performance with some fully supervised methods. We summarize the main contributions as follows:

- We propose a novel two-stage weakly supervised segmentation framework to exploit spatial and temporal information across frames. The first stage (seed point propagation) generates seeding points in different frames based on weak annotations (0.001% annotated points). The second stage (dense scene propagation) propagates high-confident points in both temporal and spatial dimensions.
- We propose a temporal propagation module using temporal matching to propagate pseudo labels to different frames. There are two matching strategies, greedy matching and optimal transport matching to search the points from the annotated objects in different frames.
- We develop a spatial graph propagation module to propagate pseudo labels along spatial dimension in the dense scene propagation stage. Spatial graph propagation generates dense pseudo labels to further improve the model.
- Experimental results on both SemanticKITTI and SemanticPOSS show that our weakly supervised two-stage framework achieves on par performance with some existing fully supervised methods, while we only use 0.001% annotations for training. Furthermore, we evaluate our weakly supervised method with 0.005% initial annotations on SemanticKITTI, and performs close to our fully supervised backbone network.

2. Related Work

3D Point Cloud Segmentation 3D point cloud semantic segmentation is a basic scene understanding task for the robotic system. Recent papers on point-cloud segmentation can be divided into projection-based methods, point-based methods, and volumetric-based methods. Projection-based methods [2, 5, 15, 30–32] transform the 3D scenario into the 2D scenario with a projection step to avoid the huge computation costs in the 3D point cloud processing. The inference speed of projection-based methods reaches the real-time requirement. However, the performance of these projection-based methods is limited by the distortion of objects and the sparsity of projected pixels in the projection step. Point-based methods directly process points to extract the spatial information. Typical point-based methods [12, 21, 22, 35] capture k nearest neighbours of each point and apply an MLP to extract the features of the point clouds. To further improve the capability of the simple MLP approach, point convolution methods [14, 16, 25] design a convolution-style operation based on the related position of the neighbours. Furthermore, some graph-based methods [28] build k nearest neighbour graph on the point cloud and adapt a graph convolution network to aggregate the features. In point-based methods, directly processing the points in the continuous space captures the original geometric information, while point-based methods require massive computation resources. Volumetric-based methods [3, 4, 9, 10, 23, 24] index the points with discrete coordinates and apply the convolution on the indexed points. With the index, volumetric-based methods accelerate the convolution on the sparse point cloud and show good performance on the large scale point cloud, *e.g.*, outdoor point cloud.

Weakly Supervised Point Cloud Segmentation Weakly supervised point cloud segmentation aims to train a usable segmentation model with weak annotations. The Weakly semantic Segmentation on indoor 3D point cloud has made great progress. Recently, MPRM [29] design a multi-Path region mining module to generate the scene-level annotation and subcloud-level annotation. Xun [33] and OTOC [18] annotate a tiny subset of indoor point cloud scene. The tiny subset contains less than 10% points for Xun [33] and 0.01% points for OTOC [18] Then, they design a self training mechanism to propagate the annotations to the whole point cloud scene, and approach the performance of fully supervised segmentation.

3. Methodology

3.1. Overview

Our framework combines spatial and temporal information to reduce the annotation costs for outdoor LiDAR point cloud datasets. An outdoor LiDAR dataset [1, 19] contains several sequences of 3D point clouds. To maximize the usage of spatial and temporal information, we split each sequence of 3D point cloud into several sub-sequences with 100 point cloud frames each. As the LiDAR devices in KITTI [8] collect ten frames per second, each of our generated sub-sequence covers data within a time range of 10 seconds. Then, we only annotate the first frame of each sub-sequence. A KITTI-style LiDAR point cloud contains target objects (e.g. car, person) and environment objects (e.g. road, building). We annotate 1 point $\mathbf{p}^t_i$ per each target object, and 20 points per each environment object. The average proportion of annotated points is around 0.001% of the whole dataset. However, only 0.001% points are not enough to train a satisfied model. We thus apply an unsupervised super-voxel segmentation on the point cloud scene and assign the same label for each super-voxel $\mathbf{v}^t_{i'}$ containing annotated points to generate more initial annotations. The unsupervised super-voxel segmentation is implemented by Lin [17], which is a simple but effective approach. The updated annotation covers 0.0057% of points in the SemanticKITTI.

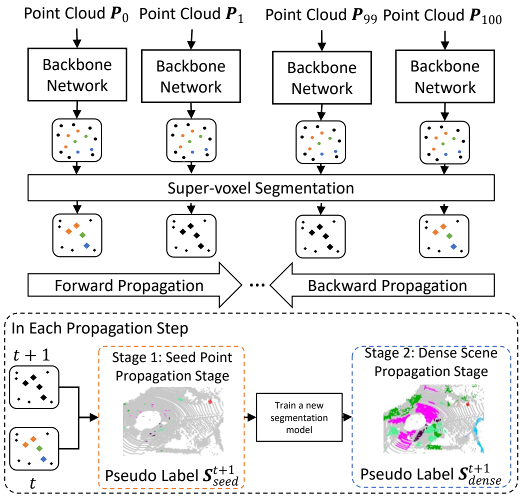


Figure 2. **The overview of our proposed framework.** Initially, we sample and annotate about 0.001% points, and train an initial model. In the seed Point propagation stage, we only select a low amount of high quality pseudo labels with temporal matching to train the first model. The dense scene propagation stage generates more data for training the final model.

With the initial annotation, one core challenges is the extreme data imbalance issue in the outdoor scenario among the target and environment objects. The proportion of annotated target object points is lower than 1% of the initial annotation. Therefore, we design a two-stage framework to improve the performance of the model. The first stage, *i.e.*, seed point propagation, uses a temporal matching with greedy matching or optimal transport [26] to search the corresponding points of the initial annotations in different point cloud frames. The updated pseudo dataset contains a low amount of pseudo labels with high quality. We train a new segmentation model with higher feature quality for the next stage with the pseudo labels from the first stage, as stated in Section 3.4. Afterwards, in the dense scene propagation stage, we use the new segmentation model to extract features. With the new features, we combine the temporal matching and spatial graph propagation to update the prediction scores of non-annotated points. The updated points with high confidence scores are the pseudo labels $\mathbf{S}_{dense}$ of the second stage. The pseudo dataset in the second stage contains a high amount of points with lower quality than the pseudo labels from the first stage. Subsequently, we con-

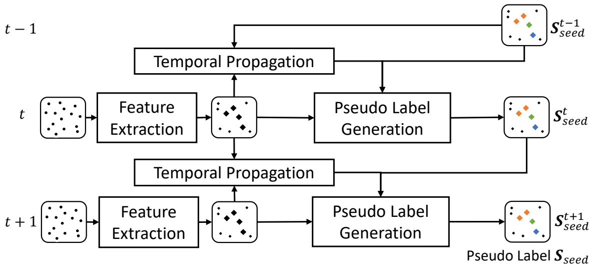


Figure 3. **The structure for the seed point propagation (SPP) stage.** Our temporal matching generates the one-to-one matching results for the super-voxel from the same objects of the pseudo labels.

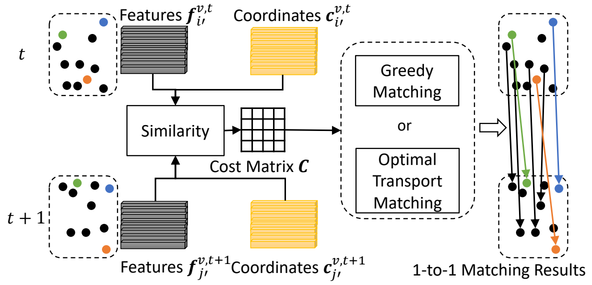

Features $f_{i'}^{v,t}$
Coordinates $c_{i'}^{v,t}$
t
Greedy Matching
or
Optimal Transport Matching
Similarity
Cost Matrix C
$t+1$
Features $f_{j'}^{v,t+1}$
Coordinates $c_{j'}^{v,t+1}$
1-to-1 Matching Results

Figure 4. **The temporal propagation module using Greedy matching (Temp-GM) or Optimal Transport (Temp-OT).**

tinue training the model from the previous stage. The two-stage framework updates the pseudo dataset with high robustness and achieves performance improvement. We show the structure of our proposed framework in Figure 2. Note that to use the limited initial annotations further, we propagate the initial annotations in both forward (from t_0 to $t_0 + 50$) and backward (from t_0 to $t_0 - 50$) directions.

3.2. Stage1: Seed Point Propagation (SPP)

As the amount of annotated data is meagre and an outdoor point cloud scene only contains coordinates of points, features obtained from pre-trained models are not reliable for updating the pseudo labels on the whole point cloud scene directly, like Xun [33] and OTOC [18]. Therefore, to solve the cold-start problems, we design a temporal matching to efficiently search the super-voxels from the same objects of annotated super-voxels in corresponding point clouds. We show the structure of the seed point propagation stage in Figure 3. The temporal matching compares the difference of features and coordinates between the points in two corresponding frames and search for the matching results of pseudo labels. The temporal matching reduces the effect of data unbalance, and is robust with just features and coordinates. In our implementation, there are two matching strategies, greedy matching and optimal transport matching. Greedy matching generates matching results with a well-designed similarity score. To further improve the performance of temporal matching, we design an optimal transport matching to generate 1-to-1 matching results with a optimal transport solver. Inspired by the point cloud flow methods [20], optimal transport builds up connections of points in two corresponding point cloud frames. The 1-to-1 matching in the optimal transport improves the performance of the model slightly. However, the optimal transport solver also increases the cost of computation. Accordingly, the choice of the matching strategies depends on the balance of performance and inference speed. We show an explanation of temporal matching in Figure 4.

Temporal matching with greedy matching (Temp-GM) As an outdoor point cloud scene covers an average of 120,000 points, directly applying the temporal matching on the original point cloud requires a massive amount of computation. Therefore, we use the results of super-voxel segmentation and update the pseudo labels at the super-voxel level. The feature $\mathbf{f}_{i'}^{v,t}$, coordinate $\mathbf{c}_{i'}^{v,t}$, and probability $\mathbf{y}_{i'}^{v,t}$ of the i'-th super-voxel $\mathbf{v}_{i'}^{t}$ are

$$\mathbf{c}_{i'}^{v,t} = \frac{1}{n'}\sum_{\hat{i}}^{n'} \mathbf{c}_{\hat{i}}^{t}; \quad \mathbf{f}_{i'}^{v,t} = \frac{1}{n'}\sum_{\hat{i}}^{n'} \mathbf{f}_{\hat{i}}^{t}; \quad \mathbf{y}_{i'}^{v,t} = \frac{1}{n'}\sum_{\hat{i}}^{n'} \mathbf{y}_{\hat{i}}^{t}, \tag{1}$$

where $\hat{i}$ represents the $\hat{i}$-th point belonging to the super-voxel $\mathbf{v}_{i'}^{t}$, and n' is the total number of points in one super-voxel. $\mathbf{c}_{\hat{i}}^{t}$, $\mathbf{f}_{\hat{i}}^{t}$, and $\mathbf{y}_{\hat{i}}^{t}$ are the coordinate, feature and probability of point $\mathbf{p}_{\hat{i}}^{t}$. The label $l_{i'}^{v,t}$ of $\mathbf{v}_{t,i'}$ is the label with the maximum probability score. Then, we build the transport cost matrix $\mathbf{C}^{t,t+1}$ to solve the optimal transport problem. In our task, the information of an outdoor point cloud contains coordinates, remissions and features from the pre-trained model. Therefore, we use the features from the pre-trained network and the coordinates to extract the matching points between $\mathbf{P}^{t}$ and $\mathbf{P}^{t+1}$. Initially, the feature similarity scores $d_{i',j'}^{f,t,t+1}$ and coordinate similarity scores $d_{i',j'}^{c,t,t+1}$ between $\mathbf{v}_{t,i'}$ and $\mathbf{v}_{t+1,j'}$are formulated as:

$$d_{i',j'}^{f,t,t+1} = \frac{(\mathbf{f}_{i'}^{v,t})^{T} \cdot \mathbf{f}_{j'}^{v,t+1}}{\left\|\mathbf{f}'_{t,i'}\right\| \cdot \left\|\mathbf{f}_{j'}^{v,t+1}\right\|}, \quad d_{i',j'}^{c,t,t+1} = \exp\left(-\frac{\left\|\mathbf{c}_{i'}^{v,t} - \mathbf{c}_{j'}^{v,t+1}\right\|^{2}}{2\theta^{2}}\right). \tag{2}$$

Note that θ is a hyperparameter, which is set to 0.5 in our implementation. We use cosine similarity to determine $d_{i',j'}^{f,t,t+1}$, which is better than the Gaussian kernel in our experiment. Then, a matching cost matrix $\mathbf{C}_{i',j'}^{t,t+1}$ is

$$\mathbf{C}_{i',j'}^{t,t+1} = 2 - d_{i',j'}^{f,t,t+1} - d_{i',j'}^{c,t,t+1}. \tag{3}$$

To reduce noise data, we set $\mathbf{C}_{i',j'}^{t,t+1}$ as ∞ if the L2 distance of $\mathbf{c}_{i'}^{v,t}$ and $\mathbf{c}_{j'}^{v,t+1}$ is larger than $10m$. The matching result of $\mathbf{p}_{i'}^{t}$ are the points $\mathbf{p}_{j'}^{t+1}$ with the lowest $\mathbf{C}_{i',j'}^{t,t+1}$.

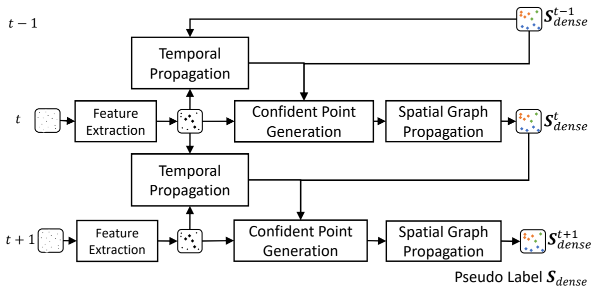


Figure 5. **The pipeline of dense scene propagation (DSP) stage.**

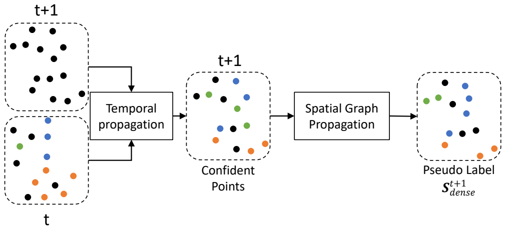


Figure 6. **Temporal matching and spatial graph propagation in the dense scene propagation stage.**

Temporal matching with Optimal Transport (Temp-OT) Optimal Transport measures a minimum cost to transport set $\mathbf{X}$ to another set $\mathbf{Y}$, and generates a transport plan $\mathbf{T}$. Based on Kantorovich's formulation [13], an optimal transport problem for our setting is formulated:

$$\begin{aligned} \mathbf{T} &= \underset{\mathbf{U}\in\mathbb{R}_{+}^{n\times m}}{\operatorname{argmin}} \sum_{i}^{n}\sum_{j}^{m} \mathbf{C}_{i,j}\mathbf{U}_{i,j}, \\ s.t. \quad & \mathbf{U}\mathbf{1}_m = \mathbf{1}n^{-1}, \mathbf{U}^T\mathbf{1}_n = \mathbf{1}m^{-1}. \end{aligned} \tag{4}$$

Here $\mathbf{C}$ is the transportation cost matrix. $\mathbf{U}$ is the assignment matrix with each element $\mathbf{U}_{i,j}$ denoting the assignment value from sample i in $\mathbf{X}$ to sample j in $\mathbf{Y}$. n and m are the amount of elements in set $\mathbf{X}$ and $\mathbf{Y}$. In our setting, $\mathbf{X}$ and $\mathbf{Y}$ are the point clouds $\mathbf{P}^t$ and $\mathbf{P}^{t+1}$. The transportation cost matrix $\mathbf{C}$ is built up on the differences of features and coordinates between $\mathbf{P}^t$ and $\mathbf{P}^{t+1}$, and we directly use the matching cost matrix $\mathbf{C}^{t,t+1}$ in Equation (3). Then, Sinkhorn algorithm [6] is capable of solving a smoothed version of optimal transport, which is formulated as:

$$\begin{aligned} \mathbf{T}^{t,t+1} &= \underset{\mathbf{U}\in\mathbb{R}_{+}^{n\times m}}{\operatorname{argmin}} \sum_{i}^{n}\sum_{j}^{m} \mathbf{C}_{i',j'}^{t,t+1}\mathbf{U}_{i,j} \\ &+ \epsilon\mathbf{U}_{i,j}(\log\mathbf{U}_{i,j} - 1), \\ s.t. \quad & \mathbf{U}\mathbf{1}_m = \mathbf{1}n^{-1}, \mathbf{U}^T\mathbf{1}_n = \mathbf{1}m^{-1}. \end{aligned} \tag{5}$$

Note that $\mathbf{T}^{t,t+1}$ is the transportation plan matrix between point cloud $\mathbf{P}^t$ and $\mathbf{P}^{t+1}$. n and m are the number of points in $\mathbf{P}^t$ and $\mathbf{P}^{t+1}$. ϵ is a hyperparameter to enhance the difference of each point pair. Then, the detailed solution is shown in Alg. 1. With the optimal transport, an one-to-one matching can be determined with $\mathbf{T}^{t,t+1}$.

Algorithm 1 Sinkhorn Algorithmn

```
Input: Transportation Cost Matrix $\mathbf{C}^{t,t+1}$, hyperparameter $\epsilon$, maximum iteration number $L_0$
Output : Transportation Plan $\mathbf{T}^{t,t+1}$
procedure
    $\mathbf{K}^{t,t+1} \leftarrow \exp(\mathbf{C}^{t,t+1}/\epsilon)$
    $\mathbf{a} \leftarrow \mathbf{1}n^{-1}, \mathbf{b} \leftarrow \mathbf{1}m^{-1}$
    for $l = 1, ..., L_0$ do
        $\mathbf{b} \leftarrow \mathbf{1}m^{-1}/((\mathbf{K}^{t,t+1})^T \cdot \mathbf{a})$
        $\mathbf{a} \leftarrow \mathbf{1}n^{-1}/(\mathbf{K}^{t,t+1} \cdot \mathbf{b})$
    end for
    $\mathbf{T}^{t,t+1} \leftarrow diag(\mathbf{a}) \cdot \mathbf{K}^{t,t+1} \cdot diag(\mathbf{b})$
end procedure
```

Updating Pseudo Label Set $\mathbf{S}_{seed}$. The matching result of $\mathbf{v}_{i'}^{t}$ is $\mathbf{v}_{j'}^{t+1}$ with the lowest matching cost or highest transport score. We assign the label of annotated points to the matching result $\mathbf{v}_{j'}^{t+1}$, and the matching results are updated as pseudo labels. Then, we apply the temporal matching sequentially to propagate the pseudo labels from the previous frame to the next frame. However, the false matching results during the propagation accumulate in the following propagation steps. Furthermore, most objects annotated in the first frame do not appear in distance frames. Therefore the one-to-one matching is not able to generate accurate matching results. Thus, the errors are propagated along with the whole process, which leads to significantly reduced quality of the generated pseudo labels. To reduce error accumulation, we use confidence scores to filter the matching results. When a prediction score of one matching result shows a low confidence score for the label of annotated point from the previous frame, we stop the propagation of this annotated point in advance. Then, we record every remaining matching result as the pseudo label set $\mathbf{S}_{seed}$. The proportion of pseudo labels $\mathbf{S}_{seed}$ is 0.8% of the whole point clouds. With $\mathbf{S}_{seed}$, we train a new model for the next stage.

3.3. Stage2: Dense Scene Propagation (DSP)

In the previous stage, temporal matching generates 0.8% pseudo labels with high quality. The proportion of pseudo labels is still low. Therefore, we propose the second stage to update more pseudo labels with dense scene propagation and continue training the model for the performance. With the model from previous stage, we firstly propagate the initial annotations along the time dimension with temporal matching to capture more pseudo labels. Then, in the spatial dimension, we use spatial graph propagation to propagate the labels of matching results to the whole point clouds and generate pseudo labels for these frames. Subsequently, we iteratively propagate the updated pseudo labels to the following frames. We show the pipeline of dense scene propagation stage in Figure 5.

Temporal Propagation Given two corresponding point clouds $\mathbf{P}^t$ and $\mathbf{P}^{t+1}$, we use the temporal matching (details given in Sec. 3.2 and Figure 4) to search the super-voxel $\mathbf{v}_{j'}^{t+1}$ from the same pseudo label of $\mathbf{v}_{i'}^{t}$. Note that $\mathbf{v}_k^{t+1}$ is the k-th super-voxel of the matching results. The matching results are the confident points in the current point cloud $\mathbf{P}^{t+1}$. For the confident points $\mathbf{v}_k^{t+1}$, we assign the one-hot labels of source super-voxel $\mathbf{v}_{i'}^{t}$ to the target super-voxel $\mathbf{v}_k^{t+1}$ as the probability $\mathbf{y}_k^{o,t+1}$.

Spatial Graph Propogation (SGP) Then, we build a directed graph $G(V,E)$ on the whole point cloud. In $G(V,E)$, the directions of edges E are from the confident points to all the super-voxels in $t+1$, which includes self-loop edges. Afterwards, we build the transition matrix $\mathbf{A}$ for graph $G(V,E)$ with similarities of super-voxels. The similarity of $\mathbf{v}_k^{t+1}$ and $\mathbf{v}_{j'}^{t+1}$ are:

$$w_{k,j'} = \exp(-\lambda_0 \frac{\left\|\mathbf{c}_k^{v,t+1} - \mathbf{c}_{j'}^{v,t+1}\right\|^2}{2\theta_0^2} - \lambda_1 \frac{\left\|\mathbf{f}_k^{v,t+1} - \mathbf{f}_{j'}^{v,t+1}\right\|^2}{2\theta_1^2}), \tag{6}$$

where λ_0, λ_1, θ_0 and θ_1 are hyper-parameters to control the weights of the features $\mathbf{c}_{j'}^{v,t+1}$ and $\mathbf{f}_{j'}^{v,t+1}$. Accordingly, we build a transition matrix $\mathbf{A}$, and each element of $\mathbf{A}$ is:

$$a_{k,j'} = \frac{w_{k,j'}}{\sum_{k'}^{m'} w_{k',j'}}, \tag{7}$$

where m' is the number of matching results. Then, we propagate the information of search results to the unannotated super-voxel. As stated in Section 3.1, there is an extreme data unbalance problem in outdoor point cloud scenario, and the information of environment objects eliminates the information of target objects during the graph propagation. To reduce the effects of data unbalance, we apply a dropout on the super-voxels of environment objects in the tail nodes. In our implementation, we only keep 5% super-voxels of environment objects for propagation. The updated probability $\hat{\mathbf{y}}_{j'}^{v,t+1}$ is:

$$\hat{\mathbf{y}}_{j'}^{v,t+1} = \alpha \mathbf{y}_{j'}^{v,t+1} + (1-\alpha)\sum_{k'}^{m'} a_{k',j'} \mathbf{y}_k^{o,t+1}, \tag{8}$$

where α is a hyperparameter. Ultimately, we select the super-voxel with high $\hat{\mathbf{y}}_{j'}^{v,t+1}$ as the pseudo label. Furthermore, when $t=0$, we directly use the annotated points as the $\mathbf{v}_{0,k}$. We show an explanation of our proposed module in Figure 6.

Updating Pseudo Label Set $\mathbf{S}_{dense}$. In the dense scene propagation stage, the predictions with high score of $\hat{\mathbf{y}}_{j'}^{v,t+1}$ are selected as new pseudo labels. Then, we merge the matching results and new pseudo labels as the Pseudo Label Set $\mathbf{S}_{dense}$, which covers an average of 20.0% points in each point cloud.

3.4. Training Pipeline

With the SPP stage in Sec. 3.2, we search the pseudo labels from the same annotated objects in different frames and train a new segmentation model for next stage. DSP stage in Sec. 3.3 extracts high-quality features using the new segmentation model. Based on the similarity of the features from the new segmentation model, the temporal matching and spatial graph propagation generate the pseudo labels of unannotated super-voxels. Finally, we train a final segmentation model using the final pseudo labels. The training pipeline is summarized in Algorithm 2. In the SPP stage, we iteratively update the pseudo labels $\mathbf{S}_{seed}$ for better performance. Then, we only update the pseudo label once in the DSP stage. In our setting, our updating mechanism depends on the first frame of each 100 frames. The false positive pseudo labels are hard to be detected and revised, leading to overfitting on those false positive pseudo labels in both updating and training phases, especially for spatial graph propagation. In our observation, iteratively updating the pseudo labels $\mathbf{S}_{dense}$ does not lead to increased performance of the final model.

Algorithm 2 Weakly Supervised 4D Point Cloud Segmentation

```
Input: Point Clouds $\mathbf{P}$
procedure
    $\mathbf{V} \leftarrow super\_voxel\_segmentation(\mathbf{P})$
    $\mathbf{S} \leftarrow sample\_and\_annotation(\mathbf{P}, \mathbf{V})$
    $InitialSegmentModel \leftarrow train(\mathbf{P}, \mathbf{S})$
    # Seed Point Propagation Stage
    for i ← 0 to 2 by 1 do
        # Update with temporal matching.
        $\mathbf{S}_{seed} \leftarrow generation\_with\_TM(\mathbf{P}, \mathbf{S})$
        $MidSegmentModel \leftarrow train(\mathbf{P}, \mathbf{S}_{seed})$
    end for
    # Dense Scene Propagation Stage
    # Update with temporal matching and spatial
graph propagation.
    $\mathbf{S}_{dense} \leftarrow generation\_with\_SGP(\mathbf{P}, \mathbf{S})$
    $FinalSegmentModel \leftarrow train(\mathbf{P}, \mathbf{S}_{dense})$
end procedure
```

4. Experiment

We evaluate our framework on the multiple scan segmentation task on SemanticKITTI [1] and SemanticPOSS [19]. In SemanticKITTI, the training set contains 9 sequences (19,130 frames). The number of frames in the validation set is 4,071 for 1 sequence, and in the testing set it is 20,351

	Stage1: SPP		Stage2: DSP			
	Temp-GM	Temp-OT	Temp-GM	Temp-OT	SGP	mIoU
Fully Sup.						60.7
OTOC† [18]						43.1
Baseline-A						40.9
Baseline-B						42.6
Model-A	√					47.7
Model-B		√				47.9
Model-C				√	√	45.4
Model-D		√	√		√	49.2
Model-E		√		√	√	**50.3**

Table 1. **Ablation study on the validation set of SemanticKITTI**. SPP: seed point propagation; DSP: dense scene propagation; Temp-GM: temporal matching with greedy matching; Temp-OT: temporal matching with optimal transport matching; SGP: spatial graph propagation. "Fully Sup." denotes the fully supervised (100% point annotations) model.

for 9 sequences. We uniformly divided 9 sequences into 198 sub-sequences with 100 frames. Note that we adjust several choices of annotated frames to make sure every sequences contain 100 frames. In the training set, we annotated 0.1% points in the first frames of the 198 sub-sequences, which leads to a total annotation budget of 0.001%. Similar to SemanticKITTI, SemanticPOSS is a LiDAR point cloud dataset of outdoor scenes. There are 6 scenes with 2988 frames in SemanticPOSS, which contain 2488 frames for training set and 500 frames for testing set. After we uniformly divided training set as 26 sub-sequences, the number of initial annotated points in SemanticPOSS is around 3000 points for the first frames of 26 sub-sequences. We implement our framework with the Minkowski Engine [4]. The whole training time for one model on SemanticKITTI is 5 days with one Nvidia RTX 3090. We set θ, ϵ, θ_0, θ_1, α and β to 0.5, 0.03, 0.5,0.3, 0.5 and 0.5, respectively.

4.1. Evaluation on SemanticKITTI

Ablation study on different components To study the effectiveness of each module, we conduct detailed experiments on the validation set of SemanticKITTI. In our experiments, all the backbones of the models are the MinkowskiUNet [4] with 42 layers. As shown in Table 1, "Fully Sup." denotes the fully supervised backbone network. Baseline-A is using only 0.001% initial annotations to train a segmentation model. Compared to "Fully Sup.", the mIoU score drops by 19.8. As comparisons, we further implement two weakly supervised methods for 3D point cloud segmentation, *i.e*., naive pseudo label generation (Baseline-B) and an existing method OTOC† [18]. OTOC is originally a weakly supervised method for indoor point cloud segmentation. The pseudo label generation of Baseline-B is to directly update the points with high confidence scores as the pseudo labels for the model training. Baseline-B performs almost comparable with OTOC† [18], improving Baseline-A by less than $\sim$2 points. This validates our claim that directly applying weakly supervised methods developed for indoor scenes to outdoor point cloud segmentation can not perform well. In contrast to OTOC, Model-B with Temp-OT achieves an absolute mIoU boost of 7.4 over Baseline-A. Our full model (Model-E) increases the mIoU score by 9.4, which is significantly better than OTOC.

	Supervision	mIoU
PointNet [21]	100%	14.6
PointNet++ [22]	100%	20.1
SqueezeSegV2 [31]	100%	39.7
DarkNet21Seg [1]	100%	47.4
KPconv [25]	100%	58.8
MinkowskiUNet [4]	100%	56.2
Baseline-100f(MinkowskiUNet)	0.001%	39.4
Ours-100f	0.001%	44.8
Baseline-20f(MinkowskiUNet)	0.005%	46.4
Ours-20f	0.005%	52.3

Table 2. **Our test results on the semantic segmentation task of SemanticKITTI.** Here, we show two results with different initial annotations. For Baseline-100f and Ours-100f, we sample 0.1% points in the first frame per 100 frames(0.001% points in total). With only 0.001% initial annotations, our model achieves comparable results with some fully supervised results [31]. For Baseline-20f and Ours-20f, we sample 0.1% points in the first frame per 20 frames(0.005% points in total). The performance of Ours-20f is only 2.9% than our fully supervised baseline.

	Supervision	mIoU
PointNet++ [22]	100%	20.1
RandLA-Net [11]	100%	53.5
KPConv [25]	100%	55.2
JS3C-Net [34]	100%	60.2
Backbone	100%	56.3
Backbone	0.001%	39.5
Backbone+SPP(Ours)	0.001%	49.4
Backbone+SPP+DSP(Ours)	0.001%	**52.2**

Table 3. **The results on the data part 3 of SemanticPOSS.** Supervision indicates the percentage of annotations in training.

We also compare our temporal matching module with greedy matching (Model-A) and an optimal transport matching (Model-B). We can see that Model-B performs slightly better than Model-A. The reason is that the proportion of generated pseudo labels is only 0.8%, and the low proportion limits the advantage of optimal transport. Further comparing Model-E with Model-D, we can see that the optimal transport matching again outperforms the greedy matching used in the DSP stage by a margin of 1.1. In DSP, the proportion of generated pseudo labels increases to around 20.0%. With more pseudo labels, Temp-OT achieves a substantial improvement of 1.1 in DSP stage.

Next we validate the two stage design of our method. The performance of our full model (Model-E) achieves an improvement of 2.4 over Model-B, which validates the ef-

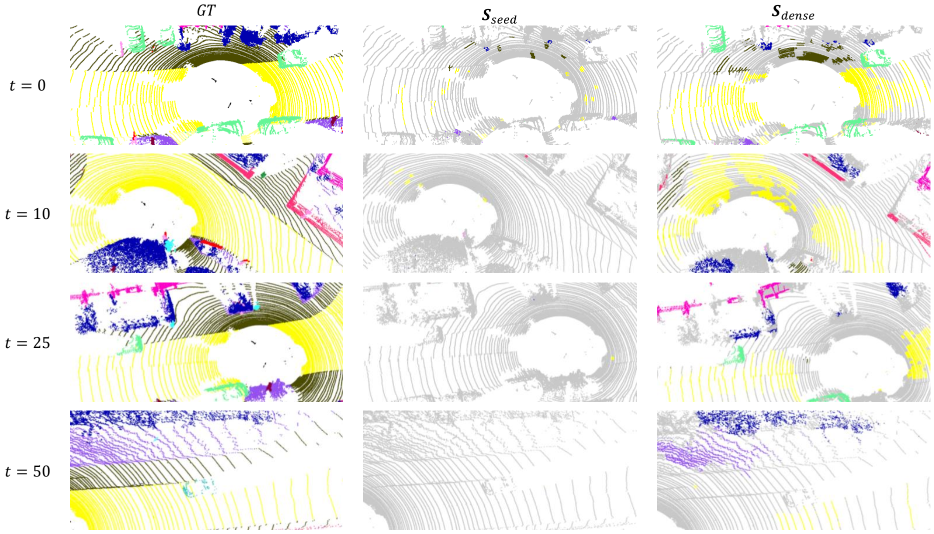


Figure 7. **Qualitative results of generated pseudo labels in stage 1 and stage 2**.

fectiveness of our DSP stage. Comparing Model-C (without SPP stage) with Model-E, we can see that adding the SPP stage leads to an mIoU boost of 4.9. This demonstrates the effectiveness of our SPP stage.

Results on the test set of SemanticKITTI Table 2 reports the results on SemanticKITTI test set. There are several representative works in fully supervised 3D semantic segmentation. These methods are the upper bounds of our weakly supervised methods and backbone network. Our baseline network is a MinkowskiUNet [4] with 42 layers, which is the same as our backbone network. Compared to the MinkowskiUNet with 100% supervision, the baseline model with 0.001% initial annotations results in an absolute mIoU drop of 16.8. While our framework outperforms the baseline with a considerable margin,bringing an absolute mIoU boost of 5.4%. Then, we evaluate the same model with 0.005% initial annotation. The performance of our baseline model with more initial annotation is 9.8% lower than fully supervised MinkowskiUNet. Our framework outperforms the our baseline model with 5.9%, and reaches the same level of performance as the fully supervised MinkowskiUNet.

The qualitative results of pseudo label. We randomly select the pseudo labels in different positions of sequences, which is shown in Figure 7. In both S_{seed} and S_{dense}, with the propagation, the number of pseudo labels in $t = 50$ is significantly lower than the number in $t = 1$, especially for S_{seed}. The frames far from the first frame do not share many annotated regions with the first frame. Therefore, building the long range connection between the annotated areas and distance frames remains a big challenge.

4.2. Evaluation on SemanticPOSS

We also evaluate our annotation and training approach on SemanticPOSS. Note that there are several target objects without any instance-level annotations. We sample 10% of the super-voxel amounts as the initial annotation of SemanticPOSS. As a result, the annotation in SemanticPOSS is denser than the annotation in SemanticKITTI. The proportion of initial annotation is still around 0.001%. As shown in Table 3, our framework achieves 52.2% on area 3 of SemanticPOSS while the model trained by the initial 0.001% annotations achieves only 39.5%.

5. Conclusion

We propose a two-stage framework to train a usable model with extremely sparse annotations (0.001% annotated points) for outdoor 3D point cloud sequences. Experimental results demonstrate that our method significantly outperforms the baseline and achieves comparable results with some fully supervised methods.

6. Acknowledgement

This study is supported under the RIE2020 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contribution from the industry partner(s). This research is also supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2018-003). This research is also supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (MOE-T2EP20220-0007) and Tier 1 (RG95/20).

References

[1] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9297–9307, 2019. 1, 2, 3, 6, 7

[2] Alexandre Boulch, Bertrand Le Saux, and Nicolas Audebert. Unstructured point cloud semantic labeling using deep segmentation networks. *3DOR*, 2:7, 2017. 2

[3] Ran Cheng, Ryan Razani, Ehsan Taghavi, Enxu Li, and Bingbing Liu. 2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12547–12556, 2021. 3

[4] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 3, 7, 8

[5] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving, 2020. 2

[6] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013. 5

[7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 2

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 3

[9] Benjamin Graham. Spatially-sparse convolutional neural networks. *arXiv preprint arXiv:1409.6070*, 2014. 3

[10] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017. 3

[11] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020. 7

[12] Mingyang Jiang, Yiran Wu, Tianqi Zhao, Zelin Zhao, and Cewu Lu. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. *arXiv preprint arXiv:1807.00652*, 2018. 3

[13] Leonid Kantorovitch. On the translocation of masses. *Management science*, 5(1):1–4, 1958. 5

[14] Deyvid Kochanov, Fatemeh Karimi Nejadasl, and Olaf Booij. Kprnet: Improving projection-based lidar semantic segmentation. *arXiv preprint arXiv:2007.12668*, 2020. 3

[15] Felix Järemo Lawin, Martin Danelljan, Patrik Tosteberg, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Deep projective 3d semantic segmentation. In *International Conference on Computer Analysis of Images and Patterns*, pages 95–107. Springer, 2017. 2

[16] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 820–830, 2018. 3

[17] Yangbin Lin, Cheng Wang, Dawei Zhai, Wei Li, and Jonathan Li. Toward better boundary preserved supervoxel segmentation for 3d point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 143:39 – 47, 2018. ISPRS Journal of Photogrammetry and Remote Sensing Theme Issue "Point Cloud Processing". 2, 3

[18] Zhengzhe Liu, Xiaojuan Qi, and Chi-Wing Fu. One thing one click: A self-training approach for weakly supervised 3d semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1726–1736, 2021. 1, 2, 3, 4, 7

[19] Yancheng Pan, Biao Gao, Jilin Mei, Sibo Geng, Chengkun Li, and Huijing Zhao. Semanticposs: A point cloud dataset with large quantity of dynamic instances. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 687–693. IEEE, 2020. 1, 2, 3, 6

[20] Gilles Puy, Alexandre Boulch, and Renaud Marlet. Flot: Scene flow on point clouds guided by optimal transport. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 527–544. Springer, 2020. 4

[21] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016. 3, 7

[22] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 3, 7

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

[24] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018. 3

[25] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. *arXiv preprint arXiv:1904.08889*, 2019. 3, 7

[26] C Villani. Optimal transport, old and new. notes for the 2005 saint-flour summer school. *Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer*, 2008. 3

[27] Haiyan Wang, Xuejian Rong, Liang Yang, Jinglun Feng, Jizhong Xiao, and Yingli Tian. Weakly supervised semantic segmentation in 3d graph-structured point clouds of wild scenes. *arXiv preprint arXiv:2004.12498*, 2020. 1

[28] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 3

[29] Jiacheng Wei, Guosheng Lin, Kim-Hui Yap, Tzu-Yi Hung, and Lihua Xie. Multi-path region mining for weakly supervised 3d semantic segmentation on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4384–4393, 2020. 1, 2, 3

[30] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893. IEEE, 2018. 2

[31] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4376–4382. IEEE, 2019. 2, 7

[32] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. *arXiv preprint arXiv:2004.01803*, 2020. 2

[33] Xun Xu and Gim Hee Lee. Weakly supervised semantic point cloud segmentation: Towards 10x fewer labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13706–13715, 2020. 1, 2, 3, 4

[34] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. *arXiv preprint arXiv:2012.03762*, 2020. 7

[35] Zhiyuan Zhang, Binh-Son Hua, and Sai-Kit Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1607–1616, 2019. 3