Weakly Supervised 3D Point Cloud Segmentation via Multi-Prototype Learning

Yongyi Su, Xun Xu Senior Member, IEEE, and Kui Jia, Member, IEEE

Abstract—Addressing the annotation challenge in 3D Point Cloud segmentation has inspired research into weakly supervised learning. Existing approaches mainly focus on exploiting manifold and pseudo-labeling to make use of large unlabeled data points. A fundamental challenge here lies in the large intra-class variations of local geometric structure, resulting in subclasses within a semantic class. In this work, we leverage this intuition and opt for maintaining an individual classifier for each subclass. Technically, we design a multi-prototype classifier, each prototype serves as the classifier weights for one subclass. To enable effective updating of multi-prototype classifier weights, we propose two constraints respectively for updating the prototypes w.r.t. all point features and for encouraging the learning of diverse prototypes. Experiments on weakly supervised 3D point cloud segmentation tasks validate the efficacy of proposed method in particular at low-label regime. Our hypothesis is also verified given the consistent discovery of semantic subclasses at no cost of additional annotations.

Index Terms—3D point cloud, weakly supervised learning, multi-prototype learning.

I. INTRODUCTION

ANNOTATING 3D point cloud for segmentation is expensive as it requires extensively annotating large number of points in 3D space and the 3D characteristics, e.g. occlusion, render annotation particularly harder. A recent approach towards tackling the annotation challenge is through learning from partially labelled data, a.k.a. weakly supervised learning [1]. This initial attempt provided insight into the mechanism of weakly supervised learning for 3D semantic segmentation using the central limit theorem. They further proposed to strengthen the task by learning geometric manifold and consistency based semi-supervised learning. Based on these insights, follow-up works are carried out by introducing propagation methods to produce better pseudo-labels as supervision [2], [3]. Despite the efforts on label propagation for more efficient use of limited labels, one inherent challenge in 3D point cloud segmentation, the large intra-class variation, remains unnoticed. For example, as illustrated in Fig. 1, there is a substantial diversity of visual appearance for “plane body”, “lampshade” and “table surface” even if they respectively represent a single semantic category. This intra-class variation results in subclasses clearly identified within each semantic category. When a linear classifier with cross-entropy loss is applied, data points with the same label are forced to group together and center around a prototype which is the weight of classifier [4]. As a consequence, this requires a very complicated representation function to map data points with varying appearance to a single point in the feature space. However, when labeled data is extremely low, e.g. only 1 point per category, training a single linear classifier (prototype) for each category is prone to underfitting [5].

Instead of having a single classifier/prototype for each semantic class, keeping multiple prototypes [5] has been adopted for few-shot learning to address multi-modal distribution within a semantic class. In this related problem, IMP [5] dynamically increases the number of prototypes following a Chinese restaurant process (CRP), when data points are far enough (above a threshold) to existing prototypes a new cluster center (prototype) is created. Despite the flexibility in

Fig. 1. Illustration of the subclass concept. Colored points indicate the activated prototypes, a clear subclass structure is observed from shape part categories, e.g. “lampshade” of pendants and lamps standing on the ground, “plane body” of fighters and passenger jets and “table surface” of square and round tables.
determining the number of prototypes, CRP is essentially a sequential process and is only effective with a small support set in few-shot learning. Generalizing CRP to weakly supervised learning, where a large number of data (up to millions of points in a single mini-batch) determines the prototypes, is subject to prohibitive computation cost. An alternative approach towards learning multiple prototypes [6] explored a seed-clustering paradigm where prototypes are generated by sampling seed points and clustering the rest points w.r.t. the seeds. Nevertheless, the seed-clustering approach requires updating prototypes in every training iteration thus causing non-convergence when integrated into end-to-end training.

In light of these challenge, we propose to introduce a multi-prototype classifier for weakly supervised 3D point cloud segmentation, termed MulPro for simplicity. In specific, we first design a multi-prototype memory bank to store the prototypes for each semantic class and each prototype would represent one subclass. In contrast to the offline prototype updating with K-means clustering [7] or moving average update adopted by [2], our design does not introduce any non-differentiable operations between prototypes and loss functions, thus enables end-to-end training.

With the introduction of multi-prototype memory bank, a new challenge arises as how to effectively train the prototypes. One naive way would be directly backpropagating from cross-entropy loss on labeled data. However, this will only provide very sparse supervision signal when label data is few. To tackle this issue, we propose to use both labeled and unlabeled data. Given the assignment of points to a prototype we enforce the prototype to be close to the feature of all assigned points, resembling taking the average, thus we also name this constraint as subclass averaging constraint. Subclass averaging is differentiable w.r.t. prototypes and can be used for gradient-base updating of prototypes. This whole process can be interpreted as nesting K-means clustering within the classifier, the forward pass computes the assignment and cluster centers are updated by subclass averaging in the backward update. Both steps can be efficiently conducted in a single forward-backward iteration.

We further notice that the multi-prototype update through subclass averaging does not prevent degenerate prototypes, e.g. some or all prototypes in one class become identical. Such a solution will cause random activation of prototypes, which is harmful for gradient-based updating. Therefore, we further propose to force the prototypes to be diverse within a category. This is achieved by penalizing the similarity between prototypes and loss functions, thus enables end-to-end training.

We propose a subclass averaging constraint to exploit both labeled and unlabeled data to supervise prototypes learning. This can be interpreted as nesting K-means clustering within the classifier. We further propose additional constraints to encourage diversity between prototypes to avoid degenerate solutions.

We improve weakly supervised 3D point cloud segmentation tasks and simultaneously discover subclasses within each semantic category without any additional supervision.

II. RELATED WORK

A. 3D Point Cloud Segmentation

Segmentation is a fundamental task in understanding 3D environment. The recent surge of deep learning methods for 3D point cloud is attributed to PointNet [8] which adapted a MLP to learn keypoints for point cloud understanding. The follow-up work, PointNet++ [9], proposed to embed PointNet in a small neighborhood to capture local geometric information. Along this line of works, convolution based methods [10], [11], [12], [13], Graph convolution based methods [14], [15], [16], [17], quadratic point-to-surface representation [18] and semantic affinity [19] are subsequently proposed to further expand the receptive field, learn better geometric representation or capture label dependencies to improve 3D point cloud segmentation. More recently, transformers [20], [21] are adapted to 3D point cloud, achieving unprecedented performance. Some large-scale point cloud semantic segmentation methods [22], [23], [24], [25] are proposed to improve semantic segmentation under real-world large-scale scenes. A more detailed review of point cloud understanding can be found in [26]. Nevertheless, the success of 3D point cloud segmentation is mainly attributed to training on large amount of labeled data. While labeling on 3D data for segmentation is particularly expensive and it has inspired works addressing weakly-supervised learning for 3D point cloud segmentation.

B. 3D Point Cloud Weakly Supervised Learning

Annotating 3D data for segmentation is expensive due to the high degree of freedom and articulated boundaries. Weakly supervised learning addresses this issue by exploiting sparsely labeled data [1], [2], [3], [27], [28], [29]. Among different definitions of weakly supervised 3D point cloud learning, [28] proposed an inexact annotation scheme by providing multiple binary labels to one region, class activation map (CAM) [30] is employed to exploit these labels to infer point-wise predictions. In another line of research, [1] assumes only a fraction of points are uniformly selected and labeled. They proved that under the i.i.d. assumption, the weakly supervised learning gradients will approximate the fully supervised ones. Inspired by the discovery made in [1], a pseudo-labeling approach [2] further improved label propagation to better exploit the unlabeled points. In [2], a per-region annotation assumption is adopted which exploits the unique information provided by ScanNet dataset [31]. The strong assumption
makes [2] restrictive to particular datasets where perfect super-point region is provided. Along this line of works, weakly supervised 3d scene segmentation with region-level boundary awareness and instance discrimination [32] proposed region-level energy-based loss and multi-stage region-level semantic contrastive learning strategy to achieve region-level boundary awareness. Dual Adaptive Transformations [33] performs local and region adaptive augmentations on point cloud to achieve better label propagation. To enable weakly supervised segmentation on large-scale point cloud scenes, [38] constructs a pretext task, i.e. point cloud colorization, to improve the representation capability of the weakly supervised network. Reference [35] pre-segments points for high efficient use of weak labels and demonstrated on autonomous driving dataset. SQN [36] queries a subset of latent representation of points within a local neighborhood to predict the consistent semantic label, so that it is able to fully utilize the sparse labels. HybridCR [37] proposed hybrid contrastive regularization, consisting of local guidance contrastive regularization and global guidance contrastive regularization, to propagate labels for large-scale point cloud scenes. Alternative to exploiting weakly labeled points, an active learning approach towards point cloud data [38] is investigated to select most informative points or super-points to annotate. PointContrast [39] proposed an unsupervised contrastive pretraining approach to adjust model parameters on large unlabeled data. Finetuning on small label data demonstrates promising results on label-efficient learning. For more generic label-efficient point cloud learning, Self-supervision was explored for label-efficient point cloud representation learning within convex decomposed parts [40]. As opposed to pseudo labeling and self-supervised pretraining, in this work, we are motivated by the large intra-class variation and concluded that having multiple prototypes/classifiers for each semantic class could alleviate the difficulty in representation learning.

C. Multi-Prototype Learning

Discovering prototypes for semantic classes has been widely adopted to support few-shot learning [41], [42], [43], zero-shot learning [44], etc. Originated in few-shot learning, multi-prototype learning aims to address the challenging of fitting prototypical network [41], [45] for multi-modal data distribution [5], [6], [46] by learning prototypes for recognizing classes with few training examples. The first attempt, IMP [5], proposed to adaptively expand prototype pool following a Chinese restaurant process which sequentially processes data points. Prototypes are estimated by an EM algorithm to model the mixed distribution to improve semantic segmentation [45]. Reference [46] proposes a k-means extension of Prototypical Networks. Despite the success in few-shot learning, it is impractical to trivially apply the sequential IMP to weakly supervised learning due to computation cost while other offline methods prevent end-to-end training of prototypes. In this work, we develop a multi-prototype memory bank to capture the subclass structure and the proposed constraints allow effective multi-prototype training. In the context of 3D point cloud deep learning, exploiting multi-prototype was briefly mentioned in [6] which generates multiple prototypes by farthest point sampling on the embedding space for support set. Compared with [6], we provide the first in-depth analysis into multi-prototype in weakly supervised 3D point cloud segmentation. The key novelty of the proposed multi-prototype learning approach is how to effectively learn non-trivial multiple prototypes and demonstrating its existence. Without the introduced class averaging constraint and diversity constraints, as shown in the ablation, multi-prototype would not be effective. In contrast, multi-prototype is implemented as clustering support points and no concrete evidence of the existence of multiple prototypes is provided in [6]. Moreover, the sample-clustering approach adopted by [6] prohibits the convergence when integrated into end-to-end training.

III. METHODOLOGY

We introduce a multi-prototype classifier, MulPro, to exploit sparse annotations. In this section, we first formally define the weakly supervised segmentation task. Then, we describe how the multi-prototype classifier works in the weakly supervised model. Finally, in view of the difficulties in learning multiple prototypes we propose two constraints to further constrain the prototype representation learning.

A. Architecture Overview

To formally define the weakly-supervised 3D point segmentation task, we follow the settings proposed in [1]. In specific, a training dataset \(D_t = \{X_i, Y_i, M_i\}_{i=1\cdots N_t}\) is provided, where \(X_i \in \mathbb{R}^{C \times N_t}\) are the \(N_t\) input points each with \(C\) dimension feature, e.g. 3D coordinates with RGB color if available, \(Y_i \in \{0,1\}^{K \times N_t}\) is the one-hot per-point segmentation label (\(K\) categories) and \(M_i \in \{0,1\}^N\) is a binary mask indicating whether ground-truth label is available. An encoder network \(Z = f(X; \Theta)\) maps input points into a \(D\) dimension feature space, \(Z \in \mathbb{R}^{D \times N}\). A classifier \(h(Z; \Omega) \in \mathbb{R}^{K \times N}\) maps encoded features into logits in segmentation category space. The existing approaches [1], [2] often define cross-entropy loss on classifier outputs and additional regularization may derive from manifold [1], pseudo-labeling [2], etc. In contrast to the above approaches, we propose to improve the classifier layer by introducing multiple prototypes to exploit the underlying subclass structures. In specific, the linear classifier \(h(Z; \Omega)\) can be expressed as below if bias is removed,

\[
h(Z; \Omega) = \Omega^T Z, \quad s.t. \quad \Omega \in \mathbb{R}^{D \times K} \tag{1}
\]

Training the linear classifier with cross-entropy loss can be seen as discovering \(K\) prototypes, each represented as \(\omega_k \in \mathbb{R}^D\), in the encoded feature space such that points belonging to the same category group together and center around the prototype, while pushing different categories away [4]. With such a design, it is assumed that a single prototype \(\omega_k\) is discovered for each semantic category. However, observing subclasses within each semantic category, the single prototype assumption could be too strong and potentially increases the risk of underfitting when small labeled data is available. To tackle this issue, we introduce a multi-prototype memory bank which maintains multiple prototypes for each class and
Fig. 2. An overview of MulPro for weakly supervised learning. With multiple subclass prototypes per class (two stars and two triangles represent different types of seats and backrests respectively) data points are classified by the closest prototype. Subclass averaging and prototype diversity constraints are employed to learn multi-prototypes effectively.

B. Multi-Prototype Classifier

Instead of maintaining one prototype per semantic category, we define a multi-prototype memory bank as

\[ \Omega \in \mathbb{R}^{D \times M \times K} \]

where the second dimension indexes \( M \) prototypes per semantic class. In the forward pass, given encoded feature \( Z \), we first take the inner product with all prototypes and this results in an attention map \( A \in \mathbb{R}^{N \times M \times K} \) as below.

\[
a_{nkm} = \sum_d z_{dn} \cdot \omega_{dmk} \quad (2)
\]

We notice that when \( M = 1 \) the multi-prototype classifier simply degenerates to standard linear classifier. With \( M > 1 \), we apply a maxpooling operation on the attention map \( A \) and this yields the classification logits \( l_{kn} = \max_m a_{nkm} \). The logits are eventually used for calculating the cross-entropy loss as,

\[
L_{CE} = -\frac{1}{N} \sum_n \sum_k y_{kn} \log \left( \sum_m \exp(l_{kn}) \right) \quad (3)
\]

Discussion: We provide a few insights for the multi-prototype design here. First, in the forward pass, multi-prototype classifier acts like a \( K \times M \) way classifier, while the linear classifier is a \( K \)-way classifier. Each data point feature is evaluated against all prototypes through Eq. (2). Instead of directly classifying into one of the \( K \times M \) classes, to respect the label ground-truth being \( K \)-way, the multi-prototype classifier first reduces the \( K \times M \)-way prediction into \( K \)-way prediction through maxpooling over prototypes (the \( M \) dimension). As a result, \( K \times M \) prototypes can be learned with only \( K \)-way labels provided. Because of the subclass structures, through this design we can identify the subclasses (up to \( M \)) as training goes on. Each prototype will naturally represent the subclass center.

C. Multi-Prototype Updating

We now elaborate how the multi-prototypes are updated. We first denote the activated prototype \( \omega_{\hat{m}\hat{k}} \) as the following, which is the prototype that returns the highest activation (inner production with encoded feature).

\[
\hat{m} n \hat{k} n \in \mathbb{R}^{D \times s \times \hat{k} n}, \hat{m} n = \arg \max_k a_{nkm} \quad (4)
\]

Given an activated prototype \( \omega_{\hat{m}\hat{k}} \), one could easily verify that the classification logit can be written as,

\[
l_{kn} = \omega_{\hat{m}\hat{k}}^T z_n \quad (5)
\]

Therefore, each activated prototype can be directly updated by backpropagating from cross-entropy loss defined in Eq. (3). Nevertheless, sparse labelled data provides weak supervision over prototypes and additional regularization is necessary for learning high quality prototypes. To this end, we further propose two constraints for training prototypes, namely subclass averaging constraint and prototype diversity constraint.

1) Subclass Averaging Constraint: The existing weakly supervised approaches update prototypes via exponential moving averaging over labeled data points [2]. As a result, it prohibits learning from unlabeled data points. In this section,
we introduce a differentiable loss on all available data to provide supervision signal in addition to the cross-entropy loss.

Specifically, we first identify the per-class activated prototype for each as \( \Omega_{k} = [\omega_{k1}; \cdots; \omega_{km}] \). These prototypes are further stacked as,

\[
\tilde{\Omega} = [\Omega_{k1}; \cdots; \Omega_{kn}] \in \mathbb{R}^{D \times M \times N}
\]

The subclass averaging constraint \( L_{\text{avg}} \) is then implemented as in Alg. 1. This algorithm accomplishes two tasks: activating the prototypes and updating the activated prototypes. We use a threshold \( \gamma \) to determine whether the data point feature belongs to one prototype. If one data point feature is similar to a prototype, above the threshold \( \gamma \), \( L_{\text{avg}} \) pulls them closer. Otherwise, we use it to update (activate) the farthest prototype. Therefore, the cosine distance \( 1 - \gamma \) represents the radius of a hypersphere covering the data points that are used to update the prototype in the center. More empirical analysis into the subclass averaging constraint design is presented in Sect. IV-E.4.

**Algorithm 1 Subclass Averaging Constraint Algorithm**

input : Point-wise encoded features \( Z \in \mathbb{R}^{D \times N} \).
Prototypes within the category of \( \omega_{n} \) as \( \tilde{\Omega} \in \mathbb{R}^{D \times M \times N} \)

output: Subclass averaging loss \( L_{\text{avg}} \)
calculate cosine similarity between point-wise features and the prototypes as \( \tilde{S} \in \mathbb{R}^{N \times M} \), \( s_{nm} = \frac{\omega_{nm} \cdot z_{n}}{\|z_{m}||z_{n}\|} \)
initialize a \( \theta \)-matrix \( \tilde{W} \in [0]^{N \times M} \).

\[
L_{\text{avg}} = \sum_{n} \sum_{m} w_{nm} \cdot s_{nm}
\]

Discussion: The proposed subclass averaging constraint can be interpreted as using data point features to update prototypes. Compared with moving average update adopted in [2], our design is superior in two ways. First, it is differentiable and can be combined with other learning objectives for an end-to-end training, while moving average update prevents combination with other losses to update the prototypes. Second, it allows using all data points, both labeled and unlabeled, to update prototypes. This enables discovering subclasses from all available data. Finally, since cosine similarity is agnostic to scale, minimizing \( L_{\text{avg}} \) will not result in an explosion of scale. Generating multiple prototypes was also mentioned under few-shot 3D point cloud [6], where multiple prototypes are generated by sampling a subset of multiple seed points using the farthest point sampling based on the embedding space. The main differences in our MulPro are that, the prototypes in our work are maintained as the parameters of the classifier and optimized by the “subclass averaging constraint” and other constraint losses, while the multi-prototypes in [6] are generated by sampling and only available for the current batch of data.

2) Prototype Diversity Constraint: Empirical results from the experiment suggest that directly updating the multi-prototype with cross-entropy loss and subclass averaging constraint does not necessarily guarantee all subclasses being discovered. In another words, there is a risk of all prototypes within a semantic class collapsing into an identical one. To avoid the collapsing issue, we further propose prototype diversity constraints to encourage diverse prototypes being discovered.

a) Prototype diversity within a semantic class: First of all, to encourage more diverse prototypes within a semantic class we penalize the accumulative similarity between prototypes as,

\[
L_{pd} = \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{j=1}^{M} \text{sim}(\omega_{m,k}, \omega_{m,j,k})
\]

For the selection of similarity metric \( \text{sim} \), we take the following considerations into account. First, the similarity should avoid any trivial solution. Therefore, any unconstrained similarity metrics should be excluded, e.g. inner product could result in a vanish of scale. Moreover, the diversity should not be overly emphasized, otherwise all prototypes could become equally distanced, and they no longer characterize the subclasses within a single class. As a result, we propose to adopt a piece-wise similarity function, specifically the following thresholded cosine similarity is adopted,

\[
\text{sim} = \max(0, \frac{\omega_{m,k}^{T} \omega_{m,k}^{T}}{\|\omega_{m,k}||\omega_{m,k}\|} - \sigma)
\]

This indicates prototype diversity within a semantic class is encouraged only when the similarity is above a threshold \( \sigma \) and the normalized cosine similarity avoids scale vanishing.

b) Balancing prototype diversity and class separability: As discussed, overly emphasizing prototype diversity could result in equally distanced prototypes and harm the separability in semantic classes. To address this issue, we introduce a metric learning loss [47] to apply further constraints to the distribution of prototypes. We first denote the per-class mean prototype \( \hat{\omega}_{k} \) as the average of all normalized prototypes within each semantic class, i.e. \( \hat{\omega}_{k} = \frac{1}{N} \sum_{n} \omega_{nk} / ||\omega_{nk}||_{2} \), and the scatters of prototypes as the variance, \( s_{k} = \frac{1}{N} \sum_{n} ||\omega_{nk}||_{2} - \|\hat{\omega}_{k} ||_{2} \). We define the constraint as minimizing the negative logarithm of the ratio between minimal inter-class mean prototype distance and maximal intra-class scatters as in Eq. (9).

\[
\mathcal{L}_{\text{bds}} = \log \max_{k} s_{k} \min_{k,j} \frac{||\hat{\omega}_{k}||_{2}^{2} - ||\hat{\omega}_{j}||_{2}^{2}}{||\hat{\omega}_{k}||_{2}^{2} - ||\hat{\omega}_{j}||_{2}^{2}}
\]

D. Training Strategy

Eventually, we combine cross-entropy loss, subclass averaging constraint and prototype diversity constraint to supervise
the update of multi-prototype memory bank.

\[ L = \lambda_{CE} L_{CE} + \lambda_{avg} L_{avg} + \lambda_{pd} L_{pd} + \lambda_{bds} L_{bds} \]  

(10)

We further notice that MulPro is compatible with the additional constraints and post-processing techniques introduced in [1].

IV. EXPERIMENT

In this section, we first introduce the benchmark datasets (Sect. IV-A). Next, we present details on our weakly supervised semantic segmentation experiments and compare with state-of-the-art methods (Sect. IV-B). We further provide analysis about multi-prototype (Sect. IV-C). Finally, the ablation study demonstrates the superiority of the multi-prototype classifier and the importance of the several losses that constrain the multi-prototype update (Sect. IV-E).

A. Datasets

We conduct experiments on four 3D point cloud segmentation datasets, covering both Shape-level 3D point cloud datasets and real-world 3D point cloud captured from both indoor and outdoor scenes.

ShapeNet [48] is a richly-annotated, large-scale 3D shapes dataset including 16,881 shapes, divided into 16 categories, each annotated with 50 parts. We evaluate part segmentation task on this dataset following the weakly supervised setting proposed in [1]. For each training sample, a subset of points, 10 percent of all points or one point per part, are randomly selected to be labelled. For testing, the comparison is performed using the default protocol.

PartNet [49] is a large-scale dataset of 3D objects annotated with hierarchical 3D part information, including 24 shape categories and a total of 26,671 unique objects. For part segmentation task, we choose the coarsest level annotation. The experiment setting is kept the same with [1].

S3DIS [50] is an indoor real scene dataset, which is widely used as a benchmark dataset for 3D segmentation evaluation. It is composed of 6 areas each including several rooms, e.g. office areas, educational and exhibition spaces. For scene segmentation, it has 13 semantic categories of indoor scene objects. Each point is represented with xyz coordinate and RGB value. For weakly supervised setting [1], a subset of points is uniformly labelled within each room. We choose Area 5 to be the test split.

SemanticKITTI [51] is a large-scale outdoor LiDAR dataset, which is widely used to evaluate the effectiveness of performing on outdoor real scenes. For semantic segmentation, it includes 19 semantic categories of outdoor scene objects. Each point is represented with 4-channels feature, which consists of (x, y, z, r) where x, y and z is the locations of the point and r is the reflection intensity. For weakly supervised setting, we randomly select a subset points with annotations within a frame and keep them as fixed labeled points during total training procedure. We choose 08 sequence as the validation set.

B. Weakly Supervised Semantic Segmentation

Encoder Network: On ShapeNet, PartNet and S3DIS datasets, we use the feature extraction encoder of DGCNN [14] with default parameters combined with our Multi-Prototype classifier as our network for fair comparison with previous work [1]. On SemanticKITTI dataset, we respectively utilize the KPConv [13] and RandLA-Net [22] as the backbone, and replace their classifier with our Multi-Prototype classifier (denoted as MulPro). Here we set \( \lambda_{CE} = \lambda_{avg} = \lambda_{pd} = \lambda_{bds} = 1 \), the hyper-parameter threshold \( \sigma \) in prototype diversity constraint to 0.2 or 0.8 for different experiments, the hyper-parameter \( \gamma \) in subclass averaging constraint algorithm to \( \cos^{-1}\frac{\sigma}{2} \) and the \( \tau \) to 0.1. \( M \) is set as 5 for ShapeNet and PartNet, and 10 for S3DIS to obtain the best results. On SemanticKITTI, we set \( M \) to 5 under KPConv backbone and 3 under RandLA-Net backbone to obtain the best results.

Implementation Details: For the labeling strategy in weak supervision setting, on ShapeNet, PartNet and S3DIS datasets, we choose the labeled points according to the labeling mask proposed by WeakSup which are fixed throughout the experiments. It’s worth noting that all of our experiments under the same annotation setting share the same labeling mask, which guarantees fair comparisons among the different competing methods. On SemanticKITTI, as no existing label mask is available, we randomly generate the labeling mask following the algorithm in Alg. 2 and fix it throughout all experiments. The generated labeling mask will be released in our repository. For training configurations, we train the MulPro with DGCNN as backbone for 200 epochs on ShapeNet, S3DIS and PartNet datasets and train the MulPro with RandLA-Net as backbone for around 100 epochs on SemanticKITTI dataset. The optimizer used in the experiments of ShapeNet, S3DIS and PartNet datasets is SGD with momentum 0.9, and the learning rate of SGD is 0.1 and the weight decay is 1e-4 for all experiments. In SemanticKITTI dataset, we follow the training configuration of KPConv-PyTorch repo2 and RandLA-Net-PyTorch repo3 without any modification. All experiments can be performed into a Nvidia GTX3090 card. To facilitate follow-up research, we released our code on GitHub.4

Comparisons: We compare against DGCNN trained under fully supervision setting (Ful. Sup.), and previous weakly supervised approaches (Weak. Sup.). Among weakly supervised approaches, we compare with WeakSup [1] and One Thing One Click (OTOC) [2]. For a fair comparison with OTOC, we modify the encoder of One Thing One Click with DGCNN encoder network and retain the moving average update of memory bank/prototypes. The resulting method is thus termed OTOC*. Another way to generate multiple prototypes is through randomly sampling seed points and clustering the rest points base on the seed points [6]. We embed this procedure in an end-to-end training pipeline by repeating the sample-cluster process in each iteration. However, empirical observation suggests this procedure prohibits model

1https://github.com/alex-van-xu/WeakSupPointCloudSeg
2https://github.com/HuguesTHOMAS/KPConv-PyTorch
3https://github.com/qiilhaer/RandLA-Net-pytorch
4https://github.com/Gorilla-Lab-SCUT/MulPro
convergence probably due to the constantly changing prototypes causing noisy learning gradients. Finally, we evaluate our multi-prototype classifier (MulPro) under the same settings.

Evaluation Metric: We calculate the mean Intersect over Union (mIoU) for each test sample as its evaluation metric. For ShapeNet, we present the average mIoU over all samples (SampAvg) and the average mIoU over all categories (CatAvg) which we firstly calculate the average mIoU over samples in each category. For S3DIS, we present the average mIoU over all samples (SampAvg) and annotation ratio \( r \)

Algorithm 2 Sample Labeled Points

| input | The point cloud \( X_i \in \mathbb{R}^{C \times N} \) with labels \( Y_i \in \{0, 1\}^{K \times N} \) and annotation ratio \( r \) |
|-------|--------------------------------------------------|
| output | The labeling mask \( M \in \{0, 1\}^N \) initialize as a \( \theta \)-vector: \( M = \{0\}^N \) |
| for \( k \in 1 \) to \( K \) do |
| \( N_k = \sum_n \mathbb{1}(Y_{kn} = 1) \) |
| The number of points with annotation: \( N^M_k = \lfloor N_k * r \rfloor \) |
| Randomly choose \( N^M_k \) points into labeled set: \( S_k = \{X_j | j = 1 \ldots N^M_k\} \) |
| Update the labeling mask: \( \forall X_i \in S_k: M_i = 1 \). |
| return \( M \) |

We present semantic segmentation results on SemanticKITTI benchmark in Tab. IV. We make the following observations from the results. i) With KPConv as backbone, our model outperforms the baseline method models with significant margin under both 1% or 0.1% annotation regimes. ii) Under the full supervision, the feature extractor of KPConv combined with our proposed multi-prototype classifier surpasses the original KPConv by 0.8%, which demonstrates our proposed multi-prototype classifier is effective and consistency under all settings, either 100%, 1%, 0.1% or 1pt labels, on SemanticKITTI dataset. iii) Improvement is more substantial at lower labeling regime, again suggesting the multi-prototype classifier is particularly effective when labels are sparse. iv) With RandLA-Net as backbone, MulPro again outperforms fully supervised baseline and HybridCR, suggesting the effectiveness of MulPro for large-scale semantic segmentation tasks.

2) Qualitative Results for Semantic Segmentation: We show qualitative results of point cloud segmentation and compare the segmentation quality. For S3DIS dataset, we visualize selected segmentation samples in Fig. 5. From left to right, the RGB view, ground-truth, fully supervised segmentation, WeakSup [1] segmentation and our MulPro result are visualized. In these visualization results, both MulPro and WeakSup leverages 1pt labelled points in the training stage. We observe that our results better respect the ground-truth for classes with large intra-class variation. For example, a “clutter” category (in black color) exists in S3DIS which covers multiple types of objects that do not fall into the other 12 predefined classes. Because of the multi-prototype classifier, our model is able to identify subclasses within this “clutter” category. This is reflected by the more consistent predictions for “clutter” class. In contrast, WeakSup makes more erroneous predictions on the “clutter” class. For ShapeNet, we show the segmentation results in Fig. 6. These examples again demonstrate competitive performance by our model when facing categories with large intra-class variation, such as the examples of the car, lamp and plane.

C. Multi-Prototype Classifier Analysis

Discovering Subclasses: Multi-prototype classifier is motivated by the subclass structures within a semantic class. In this section, we provide qualitative results on the subclasses discovered by MulPro. In specific, for each point we define the corresponding activated prototype following Eq. (4). Given a maximal \( M \) prototypes for each category we can thus assign each point into one of the prototypes by its activation. In Fig. 4, we selectively visualize points by the activated prototype, i.e.

3We use the code released by https://github.com/qiqihaer/RandLA-Net-pytorch
each individual color indicates one activated prototype. We are surprised to see many subclasses identified. For instance, despite a single body class is annotated for all planes, our multi-prototype classifier discovers an additional subclasses marked as red points in Fig. 4, corresponding to the tail part of body which is shared by all passenger jets but absent from fighter jets. Two types of wings are also discovered from the plane category, roughly differentiating passenger jets from fighters. Subclasses are also discovered from chairs, different back supports and seats are discovered, roughly distinguishing chairs with armrest from others. Subclasses are identified for lampshade as well, the ones in red are generally pendants while the pink ones are mostly floor lamps. The table surface again displays subclass structures with square-shaped desks being identified from round-shaped tables. The consistent and clean activation of prototypes among all semantic objects implies an obvious subclass structure in the feature space.

**D. Multi-Prototype Classifier at Higher Label Budget**

Multi-prototype captures the intra-class variance and is motivated under the weakly supervised setting. As demonstrated by many research on large-scale datasets with large intra-class variation, e.g. ImageNet, neural network is able to model the multi-modal distribution and a single prototype is enough for classification. However, when labeled data points are few the neural network would face underfitting, i.e. it fails...
TABLE IV
EVALUATIONS ON SEMANTIC KITTI (08 SEQUENCE) DATASET. WE COMPARED AGAINST FULLY SUPERVISED AND WEAKLY SUPERVISED APPROACHES UNDER 100%, 1%, 0.1% AND 1PT ANNOTATION SETTINGS. * INDICATES RESULTS REPORTED IN PUBLICATIONS AND ALL OTHER RESULTS ARE PRODUCED BY OUR OWN IMPLEMENTATION

| Backbone | Budget | Method      | cat  | bicycle | motorcycle | truck | other-vehicle | person | bicyclist | motorcyclist | road  | parking | sidewalk | other-ground | building | fence | vegetation | truck | terrain | pole | traffic-sign | CatAvg |
|----------|--------|-------------|------|---------|------------|-------|---------------|--------|-----------|-------------|-------|---------|----------|--------------|----------|-------|------------|-------|---------|------|-------------|-------|
| RandLA-Net* | 100%  | RandLA-Net  | 94.1 | 11.6 | 30.1 | 62.0 | 47.8 | 50.7 | 62.8 | 0.00 | 91.2 | 40.4 | 76.1 | 0.6 | 87.2 | 44.8 | 84.7 | 56.6 | 71.1 | 49.3 | 37.2 | 53.9 |
| HybridCR* | 100%  | RandLA-Net  | 94.1 | 15.5 | 27.0 | 60.4 | 50.9 | 50.6 | 66.2 | 0.00 | 91.9 | 42.3 | 77.2 | 1.8 | 87.7 | 45.4 | 84.8 | 60.2 | 71.2 | 50.3 | 36.5 | 53.3 |
| MulPro (Ours) | 100%  | RandLA-Net  | 93.6 | 11.1 | 24.7 | 57.7 | 44.6 | 45.1 | 62.2 | 0.00 | 91.1 | 36.7 | 75.9 | 0.9 | 86.4 | 39.0 | 83.6 | 58.8 | 71.9 | 50.6 | 33.1 | 51.9 |
| HybridCR* | 1%     | MulPro (Ours) | 93.6 | 12.6 | 29.3 | 72.0 | 46.0 | 50.4 | 70.5 | 0.00 | 91.7 | 39.9 | 76.5 | 1.2 | 86.4 | 42.1 | 85.4 | 56.4 | 74.1 | 52.2 | 34.5 | 53.4 |
| 0.1% | MulPro (Ours) | 91.8 | 6.2 | 21.5 | 54.4 | 36.0 | 32.2 | 59.1 | 0.00 | 87.6 | 32.7 | 71.2 | 2.1 | 83.5 | 33.7 | 80.9 | 51.5 | 71.2 | 37.3 | 20.3 | 46.0 |
| 1pt | KPConv [13] | KPConv (Ours) | 96.1 | 42.7 | 68.9 | 56.6 | 50.3 | 72.2 | 91.9 | 0.00 | 92.3 | 28.0 | 79.2 | 1.7 | 90.8 | 66.6 | 88.4 | 70.3 | 74.6 | 66.1 | 40.7 | 62.0 |
| KPConv (Ours) | 1%     | KPConv (Ours) | 96.3 | 38.5 | 72.0 | 60.9 | 54.4 | 71.0 | 92.2 | 0.00 | 92.3 | 34.1 | 79.9 | 3.8 | 90.6 | 66.6 | 88.6 | 70.6 | 75.3 | 67.4 | 38.7 | 62.8 |
| 0.1% | KPConv (Ours) | 95.3 | 19.7 | 46.0 | 48.4 | 42.9 | 55.3 | 81.4 | 0.1 | 91.0 | 30.7 | 77.2 | 1.6 | 89.6 | 62.4 | 87.5 | 66.2 | 74.5 | 61.2 | 32.2 | 56.0 |
| 1pt | KPConv (Ours) | 95.5 | 26.5 | 52.5 | 62.3 | 44.8 | 59.3 | 83.4 | 0.00 | 91.1 | 32.6 | 78.4 | 2.2 | 89.9 | 63.8 | 88.3 | 68.7 | 75.2 | 62.9 | 31.6 | 58.6 |

![Fig. 4. Visualization of activated prototypes (indicated by different colors) on selected samples from ShapeNet dataset. 1pt labelled points are used to train the weak supervision model.](image)

To validate the advantage of MulPro, we carry out additional experiments at 100% labeled data. The results in Tab. V clearly suggests multi-prototype is most effective at 1pt annotation.
reaches the maximum value when $M$ of available prototypes ($M$ results in Fig. 7 from which we observe that the number of classes have more activated prototypes, e.g. "airplane", "car", and the gap between DGCNN and MulPro diminishes at higher labeling regime. This observation also motivates us to explore multiple prototypes in weakly supervised setting.

### E. Ablation and Additional Study

1) **Importance of Individual Components:** We analyze the importance of the proposed modules. Different combinations of the modules are evaluated on ShapeNet benchmark dataset with 1pt annotation scheme. The results are presented in Tab. X. We notice that the prototype diversity must simultaneously incorporate Eq. (7) and Eq. (9) to avoid trivial solution and we combine them in the ablation study. From the ablation results, we first evaluate multi-prototype classifier alone, and it yields slightly worse result than baseline, suggesting multi-prototype cannot be effectively trained with cross-entropy loss alone. Then we combine multi-prototype classifier with subclass averaging loss, this improves upon multi-prototype alone, indicating multi-prototype requires more supervision than cross-entropy loss to update. Finally, we combine prototype diversity terms and demonstrate the best result. We also evaluate our results using the post-processing technique (w/ PP) proposed in [1] which implements label propagation on network prediction and observe further improvement.

2) **Compatibility With Alternative Backbone:** We further evaluate the multi-prototype classifier with state-of-the-art 3D point cloud backbone network, namely the Point Cloud Transformer (PCT) [21] and present the results at 1pt annotation scheme. The results are presented in Tab. VI. Significant improvement is observed by combining PCT with Multi-prototype classifier.

3) **Number of Multi-Prototype:** We investigate the impact of the number of multi-prototype on segmentation performance. In specific, we evaluate $M = 1 \cdots 10$ on ShapeNet segmentation with 1 point per category label. We present the results in Fig. 7 from which we observe that the number of activated prototypes increases with the increase of the number of available prototypes ($M$), however, the mean category IoU reaches the maximum value when $M = 5$.

### 4) Subclass Averaging Constraint Design: Due to the importance of subclass averaging loss, we investigate several alternative designs. First, a naive way to use both labeled and unlabeled data to update prototype is through pseudo-labeling [54]. We predict the pseudo-labels for all unlabeled data points and the pseudo-labels are in turn used to supervised cross-entropy loss on unlabeled data. Alternatively, we could use Frobenius norm to measure the distance between data points and corresponding activated prototypes. Specifically, we stack all activated prototypes over data points as, $\mathbf{\Omega} = [\omega_{n_1k_1}; \cdots ; \omega_{n_Nk_N}] \in \mathbb{R}^{D\times N}$. Then the subclass averaging loss is calculated as, $\mathcal{L}_{avg1} = ||\mathbf{\Omega} - \mathbf{Z}||_F^2$. Since the distance metric consists of Frobenius norm and inner product between $\mathbf{\Omega}$ and $\mathbf{Z}$, the Frobenius norm parts will affect the scale, it might lead to trivial solutions. To avoid these trivial solutions, we propose the second candidate, using cosine distance to perform subclass averaging loss as,

$$ \mathcal{L}_{avg2} = - \sum_n \frac{(\omega_{n_1k_i}, \mathbf{z}_n)}{||\omega_{n_1k_i}|| \cdot ||\mathbf{z}_n||} $$

(11)

Results for comparing all alternative designs are presented in Tab. VII. We make the following observations from the results. First, we show the baseline performance that directly classifies $K \times M$ classes with pseudo-labeling. As the results show, pseudo-labeling gives the worst results, probably due to the confirmation bias [55] to blame. Furthermore, using the cosine similarity to update prototypes $\mathcal{L}_{avg2}$ avoids the trivial solution and the performance is slightly better than that using L2 distance alone $\mathcal{L}_{avg1}$. Finally, our thresholded subclass averaging outperforms both candidates, suggesting it is necessary to selectively use most relevant point features to update prototypes.

We further compare the number of uniquely activated prototypes under different subclass averaging losses. We present the results on ShapeNet and S3DIS in Tab. VIII and Tab. IX respectively. We make the following observations from the results. First, the average and median numbers of activated prototypes are very few with both L2 distance and cosine distance as subclass averaging loss on both ShapeNet and S3DIS. This is probably due to the challenge in prototype initialization. When one or a few prototypes are activated, there is no force to encourage the activation of other ones. An outlier happens for the "sofa" category, where $\mathcal{L}_{avg2}$ activated 10 prototypes. This is probably due to the brittle design of the alternative subclass averaging loss function. Moreover, with our proposed subclass averaging loss, we notice certain classes have more activated prototypes, e.g. "airplane", "car".

### TABLE V

| Method       | Annotation | SampAvg(%) | CatAvg(%) |
|--------------|------------|------------|-----------|
| DGCNN        | 1pt        | 72.6       | 72.2      |
| DGCNN        | 10%        | 84.5       | 81.5      |
| DGCNN        | 100%       | 85.1       | 82.3      |
| DGCNN + MulPro | 1pt       | 79.4       | 77.8      |
| DGCNN + MulPro | 10%      | 85.3       | 82.0      |
| DGCNN + MulPro | 100%      | 85.5       | 82.4      |

### TABLE VI

| Method       | Annotation | mIoU (%)   |
|--------------|------------|------------|
| PCT [21]     | 1pt        | 41.6       |
| PCT + MulPro | 1pt        | 43.0       |
| PCT          | 100%       | 51.5       |

### TABLE VII

| Subclass Averaging Options | ShapeNet | S3DIS |
|----------------------------|---------|-------|
| Pseudo Labeling            | 74.2    | 44.4  |
| $\mathcal{L}_{CE} + \mathcal{L}_{avg1} + \mathcal{L}_{psd} + \mathcal{L}_{bsd}$ | 74.5    | 45.1  |
| $\mathcal{L}_{CE} + \mathcal{L}_{avg2} + \mathcal{L}_{psd} + \mathcal{L}_{bsd}$ | 75.8    | 45.5  |
| $\mathcal{L}_{CE} + \mathcal{L}_{avg2} + \mathcal{L}_{psd} + \mathcal{L}_{bsd}$ (Ours) | 76.4    | 46.8  |
TABLE VIII
THE NUMBERS OF ACTIVATED PROTOTYPES IN EACH SHAPE ON SHAPENET DATASET

| Subclass Averaging Options | Air. | Bag | Cap | Car | Chair | Hat | Guitar | Knife | Lamp | Lap. | Motor. | Mug | Pistol | Rocket | Skate | Table | Mean / Median |
|----------------------------|------|-----|-----|-----|-------|-----|--------|-------|------|------|--------|-----|--------|--------|-------|-------|---------------|
| L_{CR} + L_{woPA} + L_{Pd} + L_{kPS} | 1    | 1   | 1   | 1   | 1     | 1   | 1      | 1     | 1    | 1    | 1      | 1   | 1      | 1      | 1     | 1     | 1 / 1         |
| L_{CR} + L_{woPB} + L_{Pd} + L_{kPS} | 1    | 1   | 1   | 1   | 1     | 1   | 1      | 1     | 1    | 1    | 1      | 1   | 1      | 1      | 1     | 1     | 1 / 1         |
| L_{CR} + L_{woPB} + L_{Pd} + L_{kPS} (Ours) | 5    | 1   | 5   | 5   | 1     | 3   | 3      | 5     | 1    | 1    | 3      | 2   | 4      | 1      | 5     | 2.7 / 2.5     |

TABLE IX
THE NUMBERS OF ACTIVATED PROTOTYPES IN EACH SEMANTIC PART ON S3DIS (AREA 5) DATASET

| Subclass Averaging Options | ceil. | floor | wall | beam | col. | win. | door | chair | table | book | sofa | board | clutter | Mean / Median |
|----------------------------|-------|-------|------|------|------|------|------|-------|-------|------|------|-------|---------|---------------|
| L_{CR} + L_{woPB} + L_{Pd} + L_{kPS} | 1    | 1   | 5   | 1    | 1    | 2    | 1    | 1     | 2     | 1    | 1    | 6     | 1.8 / 1    |
| L_{CR} + L_{woPB} + L_{Pd} + L_{kPS} | 1    | 1   | 5   | 1    | 1    | 1    | 3    | 3     | 2     | 1    | 10   | 5     | 2.7 / 1    |
| L_{CR} + L_{woPB} + L_{Pd} + L_{kPS} (Ours) | 9    | 7   | 7   | 2    | 4    | 1    | 6    | 3     | 6     | 2    | 4    | 2     | 4.5 / 4    |

Fig. 5. Qualitative examples for S3DIS semantic segmentation. Weak Sup. refers to the results of [1].

TABLE X
ABLATION STUDY ON THE IMPACT OF INDIVIDUAL MODULES. THE RESULTS CONSIST OF WITHOUT POST PROCESSING (WO/PP) AND WITH POST PROCESSING (W/PP)

| Components | mIoU (%) |
|------------|---------|
| Multi-Proto. Subclass Avg. Proto. Diversity | w/o PP w/ PP [1] |
| - | - | 73.8 | 74.4 |
| ✓ | - | - | 73.7 |
| ✓ | ✓ | - | 75.4 |
| ✓ | ✓ | ✓ | 76.4 | 77.8 |

“chair” on ShapeNet and “ceil.”, “door”, “chair”, “table”, “clutter” on S3DIS, suggesting large intra-class variation.

TABLE XI
EVALUATION OF THE MAXIMAL NUMBER OF SUB-CLASSES M AS A HYPER-PARAMETER

| Subset | M | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|--------|---|---|---|---|---|---|---|---|---|---|----|
| Subset 1 |   | 85.44 | 80.37 | 79.34 | 80.78 | 80.83 | 80.73 | 80.85 | 80.45 | 80.46 | 79.94 |
| Subset 2 |   | 65.62 | 65.43 | 67.32 | 67.25 | 68.40 | 64.51 | 66.61 | 65.68 | 67.36 | 67.73 |
| Full Set |   | 75.03 | 73.70 | 73.33 | 74.02 | 74.62 | 74.49 | 73.81 | 73.07 | 75.92 | 75.84 |

“chair” on ShapeNet and “ceil.”, “door”, “chair”, “table”, “clutter” on S3DIS, suggesting large intra-class variation.

5. Transferability of the Number of Multi-Prototype: We split the ShapeNet into two subsets, where each subset consists of 8 shapes and around 25 parts. We respectively train MulPro on these two subsets and the full set under different M values.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
The detailed results are presented in Tab. XI and we make the observations that the Full Set performance is the best when \( M = 5 \), which coincides with the optimal value of \( M \) obtained from Subset 2. The performance on Subset 1 also achieves the second best result with \( M = 5 \). These observations suggest the choice of \( M \) depends on the variation within each semantic class and we could select the optimal \( M \) for a new dataset based on the results on a related dataset. Nevertheless, we believe that the hyper-parameter tuning on a new dataset without any prior knowledge is still a challenging problem.

V. CONCLUSION

In this work, we first observe that existing approaches towards point cloud segmentation often employ a linear classifier to separate semantic classes. This is equivalent to allocating one prototype for each category. As we point out through experiment and intuition, clear subclass structures in each semantic class of 3D point cloud segmentation data widely exist and would result in large intra-class variation in feature representation. As a result, the single prototype may not capture the large variation and could lead to inferior results. To tackle this issue, we proposed a multi-prototype memory bank where each prototype serves as the classifier for one subclass. To enable effective multi-prototype training, we further introduced two constraints. Extensive results on weakly supervised 3D point cloud segmentation benchmark suggest the advantage of maintaining multiple prototypes in particular at low-label regime. We hope the subclasses identified from ShapeNet could provide insights into future segmentation model design at low-label regime. More importantly, as the performance of weakly supervised semantic segmentation is already close to the fully supervised one, future research should focus more on demonstrating the effectiveness on a wider range of 3D scenes and objects. Finally, although subclass structures is automatically discovered from the unlabeled data, we need to specify the maximal number of sub-classes as a hyper-parameter. A careful selection of such hyper-parameter
could have a impact on the performance and automatically selecting the appropriate hyper-parameters remains as an open question.

REFERENCES

[1] 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.

[2] Z. Wu, X. Qi, and C. 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.

[3] A. Tao, Y. Duan, Y. Wei, J. Lu, and J. Zhou, “SegGroup: Seg-level supervision for 3D instance and semantic segmentation,” IEEE Trans. Image Process., vol. 31, pp. 4952–4965, 2022.

[4] Z. B. Battaglia et al., “A unifying mutual information view of metric learning: Cross-entropy vs. pairwise losses,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 548–564.

[5] R. Jiang and Z. Cheng, “Mixture Gaussian prototypes for few-shot learning,” in Proc. Int. Conf. Data Mining Workshops (ICDMW), Dec. 2021, pp. 232–241.

[6] N. Zhao, T. S. Chin, and G. H. Lee, “Few-shot 3D point cloud semantic segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 8869–8878.

[7] O. Rippel, M. Paluri, P. Dollar, and L. Bourdev, “Metric learning with adaptive density discrimination,” 2015, arXiv:1511.05939.

[8] C. R. Qi, L. Yi, H. Su, 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. 77–85.

[9] 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., vol. 30, 2017, pp. 1–11.

[10] 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., vol. 31, 2018, pp. 1–11.

[11] W. Wu, Z. Qi, and L. Fei-Fei, “PointConv: Deep convolutional networks on 3D point clouds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 9613–9622.

[12] S. Wang, S. Suo, W. Ma, A. Pokrovsky, and R. Urtasun, “Deep parametric continuous convolutional neural networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 2589–2597.

[13] H. Thomas, C. R. Qi, J. Deschaud, B. Marche, O. Litany, and N. J. Mitra, “Dual adaptive transformations for weakly supervised point cloud segmentation,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019, pp. 1554–1564.

[14] H. Ran, J. Liu, and C. Wang, “Surface representation for point clouds,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, no. 10, pp. 6955–6964, Oct. 2022.

[15] S. Liu, Y. Zhou, C. R. Qi, L. Guibas, and O. Litany, “PointConv: Flexible and deformable convolution for point clouds,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, pp. 600–619, 2020.

[16] S. Qiu, S. Anwar, and N. Barnes, “Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 1757–1767.

[17] S. Fan, Q. Dong, F. Zhu, Y. Lv, P. Ye, and F. Wang, “SFC-Net: Learning spatial contextual features for large-scale point cloud segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 14499–14508.

[18] H. Ran, J. Liu, and C. Wang, “Surface representation for point clouds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 18920–18930.

[19] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, “Deep learning for 3D point clouds: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 12, pp. 4338–4364, Dec. 2021.

[20] Q. Meng, W. Wang, T. Zhou, J. Shen, Y. Jia, and L. Van Gool, “Towards a weakly supervised framework for 3D point cloud object detection and annotation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 8, pp. 4454–4468, Aug. 2022.

[21] J. Wei, G. Lin, K. Yan, T. Hung, and L. Xie, “Multi-path region mixing for weakly supervised 3D semantic segmentation on point clouds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 4383–4392.

[22] G. Zhou, D. Wang, Y. Yan, H. Chen, and Q. Chen, “Semi-supervised 3D object pose estimation without using real annotations,” IEEE Trans. Circuits Syst. Video Technol., vol. 32, no. 8, pp. 5163–5174, Aug. 2022.

[23] 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.

[24] 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. 2432–2443.

[25] K. Liu, Y. Zhao, Q. Nie, Z. Gao, and B. M. Chen, “Weakly supervised 3D scene segmentation with region-level boundary awareness and instance discrimination,” in Proc. Eur. Conf. Comput. Vis., 2022, pp. 37–55.

[26] Z. Wu, Y. Wu, G. Lin, J. Cai, and C. Qian, “Dual adaptive transformations for weakly supervised point cloud segmentation,” in Proc. Eur. Conf. Comput. Vis., 2022, pp. 78–96.

[27] Y. Zhang, Z. Li, X. Xie, Y. Qu, C. Li, and T. Mei, “Weakly supervised semantic segmentation for large-scale point cloud,” in Proc. AAAI Conf. Artif. Intell., vol. 35, no. 4, 2021, pp. 3421–3429.

[28] M. Liu, Y. Zhou, C. R. Qi, B. Gong, H. Su, and D. Anguelov, “Less: Label-efficient semantic segmentation for LiDAR point clouds,” in Proc. Eur. Conf. Comput. Vis., 2022, pp. 70–89.

[29] Q. Hu et al., “SQN: Weakly-supervised semantic segmentation of large-scale 3D point clouds,” in Proc. Eur. Conf. Comput. Vis., 2022, pp. 600–619.

[30] M. Li et al., “HybridCR: Weakly-supervised 3D point cloud semantic segmentation via hybrid contrastive regularization,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 14910–14919.

[31] X. Shi, X. Xu, K. Chen, L. Cai, C. Sheng Foo, and K. Jia, “Label-efficient point cloud semantic segmentation: An active learning approach,” 2021, arXiv:2101.06931.

[32] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany, “PointContrast: Unsupervised pre-training for 3D point cloud understanding,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 574–591.

[33] M. Gadelha et al., “Label-efficient learning on point clouds using approximate convex decompositions,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 473–491.

[34] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” in Proc. Adv. Neural Inf. Process. Syst., vol. 30, 2017, pp. 1–11.

[35] S. Liu, M. Jiang, and J. Kong, “Multidimensional prototype refactor enhanced network for few-shot action recognition,” IEEE Trans. Circuits Syst. Video Technol., vol. 32, no. 10, pp. 6955–6966, Oct. 2022.

[36] W. Jiang, K. Huang, J. Geng, and X. Deng, “Multi-scale metric learning for few-shot learning,” IEEE Trans. Circuits Syst. Video Technol., vol. 31, no. 3, pp. 1091–1102, Mar. 2021.

[37] H. Yang et al., “Iterative class prototype calibration for transductive zero-shot learning,” IEEE Trans. Circuits Syst. Video Technol., vol. 33, no. 3, pp. 1236–1246, Mar. 2023.
[45] B. Yang, C. Liu, B. Li, J. Jiao, and Q. Ye, “Prototype mixture models for few-shot semantic segmentation,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 763–778.

[46] J. Deuschel et al., “Multi-prototype few-shot learning in histopathology,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW), Oct. 2021, pp. 620–628.

[47] X. Xu, L. Zhang, L. Cheong, Z. Li, and C. Zhu, “Learning clustering for motion segmentation,” IEEE Trans. Circuits Syst. Video Technol., vol. 32, no. 3, pp. 908–919, Mar. 2022.

[48] L. Yi et al., “A scalable active framework for region annotation in 3D shape collections,” ACM Trans. Graph., vol. 35, no. 6, pp. 1–12, Nov. 2016.

[49] A. Arvanitidis, V. Kavasche, and G. Tolias, “Label propagation for deep semi-supervised learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 5065–5074.

[50] E. Arazo, D. Ortego, P. Albert, N. E. O’Connor, and K. McGuinness, “Pseudo-labeling and confirmation bias in deep semi-supervised learning,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2020, pp. 1–8.

Yongyi Su received the B.E. degree from the South China University of Technology, Guangzhou, China, in 2021, where he is currently pursuing the Ph.D. degree with the School of Electronic and Information Engineering. His research interests include 3D weakly supervised learning, domain adaptation, test-time training, and robust learning.

Xun Xu (Senior Member, IEEE) received the B.E. degree from Sichuan University in 2010 and the Ph.D. degree from the Queen Mary University of London in 2016. He was a Research Fellow with the National University of Singapore, from 2016 to 2019. He is currently with I2R, A*STAR. His research interests include semi-supervised learning, domain adaptation, and zero-shot learning with applications to 3D point cloud data.

Kui Jia (Member, IEEE) received the B.E. degree from Northwestern Polytechnic University, Xi’an, China, in 2001, the M.E. degree from the National University of Singapore, Singapore, in 2004, and the Ph.D. degree in computer science from the Queen Mary University of London, London, U.K., in 2007. He was with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China; The Chinese University of Hong Kong, Hong Kong; the Institute of Advanced Studies, University of Illinois at Urbana-Champaign, Champaign, IL, USA; and the University of Macau, Macau, China. He is currently a Professor with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China. His current research interests include theoretical deep learning and its applications in vision and robotic problems, including deep learning of 3D data and deep transfer learning.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.