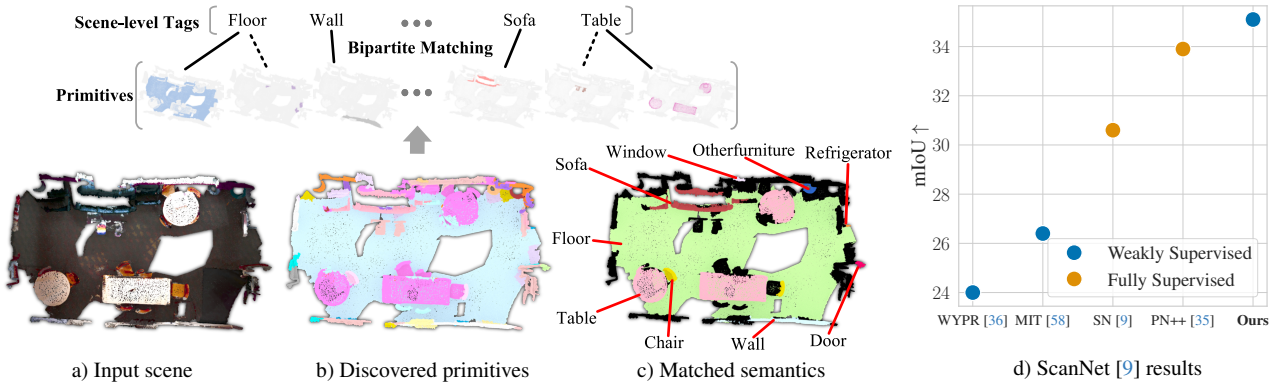
Densify Your Labels: Unsupervised Clustering with Bipartite Matching for Weakly Supervised Point Cloud Segmentation

Shaobo Xia[1*] Jun Yue[2*] Kacper Kania[3] Leyuan Fang[4]
Andrea Tagliasacchi[5,6] Kwang Moo Yi[7] Weiwei Sun[7†]
[1]Changsha University of Science and Technology [2]Central South University [3]Warsaw University of Technology
[4]Hunan University [5]Simon Fraser University [6]University of Toronto [7]University of British Columbia
https://densify-your-labels.github.io



a) Input scene b) Discovered primitives c) Matched semantics d) ScanNet [9] results

Figure 1. **Teaser** – We present a framework that can learn accurate dense, *per-point* segmentation from *scene-level* annotations. (*left*) We leverage an off-the-shelf, unsupervised method to discover primitives that govern the dataset. We group those primitives to propagate scene-level information across points. We then use bipartite matching to associate available scene-level information to discovered primitives. The method propagates the information conservatively to the most relevant clusters during training. (*right*) Our approach demonstrates a substantial performance improvement, surpassing the previous weakly supervised methods with an impressive 8.7 boost in mIoU on ScanNet [9]. Moreover, our approach achieves comparable results with certain fully supervised methods.

Abstract

We propose a weakly supervised semantic segmentation method for point clouds that predicts "per-point" labels from just "whole-scene" annotations while achieving the performance of recent fully supervised approaches. Our core idea is to propagate the scene-level labels to each point in the point cloud by creating pseudo labels in a conservative way. Specifically, we over-segment point cloud features via unsupervised clustering and associate scene-level labels with clusters through bipartite matching, thus propagating scene labels only to the most relevant clusters, leaving the rest to be guided solely via unsupervised clustering. We empirically demonstrate that over-segmentation and bipartite assignment plays a crucial role. We evaluate our method on ScanNet and S3DIS datasets, outperforming state of the art, and demonstrate that we can achieve results comparable to fully supervised methods.

1. Introduction

Semantic segmentation of point clouds is a core problem in 3D Computer Vision [12, 34, 45]. It is the foundation of many applications, including scene understanding [34], semantic reconstruction [31], and urban mapping [15]. Many state-of-the-art methods [8, 30, 49] rely on dense, *per-point* supervision which requires intense manual labor. For example, annotating ScanNet [9] took 500 workers an average of 22.3 minutes per scene, and there are 1513 scenes in the dataset (≈1 month of 24/7 annotation time). To reduce this effort, one could resort to weakly-supervised learning [14, 25, 46, 47, 50, 54], which assumes access to much

*Equal contribution.
†Corresponding author.

fewer annotations than the fully supervised counterparts.

Among weakly-supervised methods, in this work we focus on *scene-level* labels, where the training signal is the existence of a given class within the scene. Clearly, scene labels are significantly cheaper to obtain than dense ones, as each category takes only ≈1 second to annotate [32]. This means the annotation cost of a single scene in ScanNet [9] is reduced 20×, from ≈20 minutes, to roughly ≈1 minute [57]. Further, scene-level labels do not rely on specialized 3D annotation software or extensive annotator training.

While recent state-of-the-art methods for scene-level supervised point cloud segmentation differ in architecture [36, 50, 57, 58], to our best knowledge most of them rely on Class Activation Mapping (CAM) [64]. CAM supervises the segmentation using scene-level labels, bypassing the need for dense labels. However, at training time, all points within a scene share the same ground-truth scene-level label, leading to significant performance degradation as the model's task is to assign a *single* class to a point. We posit that we can actually propagate scene-level labels to points more conservatively.

To this end, we first *over-segment* the point cloud feature space via K-means [27] into *feature centroids* [62], which we call *primitives*. We then associate a subset of these primitives with the scene labels through bipartite matching [20]. Note that with bipartite matching, we are being conservative, as only the most confident primitives receive training signals from scene labels. Our method *jointly* trains computation of primitives with CAM, which we show to be essential for stable training and leads to more consistent and reliable assignments. We achieve new state-of-the-art results on both ScanNet [9] and S3DIS [2], surpassing the previous baselines respectively by 8.7 and 2.1 in mIoU. For the first time, our method outperforms certain fully supervised methods on ScanNet (see Fig. 1), highlighting the real potential of weakly supervised learning in point clouds.

In summary, in this paper:

- We introduce a new approach to propagate scene-level labels to points via primitives discovered in an unsupervised manner.
- We provide new insights into why methods relying on class activation mapping (CAM) fail to distinguish between semantic classes.
- We show how to use bipartite matching to propagate *scene-level* annotations to *per-point* labels, which significantly reduces issues existing in recent CAM methods.

2. Related works

Weakly supervised point cloud segmentation. This task aims to achieve point cloud semantic segmentation with a significantly reduced labeling cost. Most commonly, a few scenes in a dataset have full annotations, and most of the other scenes have no labels [10, 16, 19, 63]. Exploiting unannotated data's full potential is key to obtaining high-quality results [5, 16, 56]. Other works assume that only 0.1%-10% points need to be annotated [13, 21, 22, 26, 43, 54, 55, 60, 61]. Alternatively, one can annotate bounding boxes [6], scribbles [47], pre-processed segments [46] and clusters [25]. As there are direct connections between points and semantic labels, they mainly focus on propagating reliable labels to unknown regions [14, 23, 24]. For instance, Hu et al. [14] explores how labels propagate at different scales, enabling unlabeled points to utilize point-level annotations. Nevertheless, these approaches still need accurate and costly point-level labels, while weakly supervised semantic segmentation with *scene-level* annotations only requires listing the category names in a scene [50]. Scene-level annotations provide more opportunities for low-cost labeling, including leveraging cross-modal data (*e.g.* text, speech, images), and our research thus focuses on this direction.

CAM in point clouds. The critical challenge of scene-level weakly supervised learning lies in the absence of a direct connection between semantic tags and points. Class Activation Mapping (CAM), aided by global average pooling (GAP), is an essential tool to bridge per-point and scene-level tags [64]. Wei et al. [50] introduce CAM into weakly supervised point cloud segmentation. Ren et al. [36] use a bottom-up point merging algorithm to merge points within each scene into pseudo-instances, thereby improving the association between semantic objects and scene tags under CAM. Yang et al. [57] develop a CAM-based weakly supervised semantic segmentation framework based on Transformers [48]. They improve the mapping relationship by considering the proportions of different objects in a scene through a weighted GAP. Additionally, Yang et al. [58] show that super-voxels used as tokens in Transformers can further improve the semantic accuracy. However, these CAM-based approaches are known for sharing the same scene-level label between points, even though a point should be assigned to a single class only. Our work addresses this core limitation.

Unsupervised learning. Unsupervised learning can learn primitives (or parts) that contain semantic patterns without any annotations [7, 17]. In point clouds, object-level segmentation via unsupervised learning has been studied extensively [40, 44]. Several methods [13, 52] also over-segment point clouds into clusters based on the learned discriminative features, however, those clusters are error-prone as shown by Zhang et al. [62]. Zhang et al. [62] proposes a practical unsupervised method based on K-means, and achieves state-of-art unsupervised semantic segmentation for point clouds. As the key to the superior performance,

their method explores cross-scene prior by performing K-means in the entire training dataset, *i.e.*, a strategy which is extensively used in the image domain [7, 18, 51, 65]. While we leverage these advances to obtain primitives, we experimentally show that combining these with scene-level supervision is not trivial, and our method addresses this very problem.

Bipartite matching. Bipartite matching pairs elements from two distinct sets [20]. It finds the optimal subset of edges between two disjoint sets of vertices in a bipartite graph that minimizes the assignment cost. It has been used with success in recent methods for object detection [3], bounding box annotation [42], and unsupervised object discovery [28]. The optimal assignment between two sets can also be regarded as an optimal transport problem [4, 17, 39]. We use bipartite matching to align learned primitives with scene-level tags. With such a design, we achieve a new state-of-the-art in scene-level weakly supervised semantic segmentation.

3. Method

In Figure 3, we provide an overview of our weakly-supervised approach to point cloud segmentation. Our method requires only *scene-level* labels at training time that indicate whether objects of a given class exist within the given scene. Formally, given a dataset $\mathcal{P} = \{\mathbf{P}_n\}$ of N point clouds, where $\mathbf{P}_n \in \mathbb{R}^{M_n \times 3}$, and the scene-level multi-hot labels $\mathcal{Y} = \{\mathbf{y}_n\}$, where $\mathbf{y}_n \in \{0,1\}^C$ indicates whether the n-th scene contains points belonging to one (or more) of the C categories, we aim to predict dense semantic segmentation $\mathcal{C} = \{\mathbf{C}_n\}$, where $\mathbf{C}_n \in \mathbb{R}^{M_n \times C}$. We make no assumptions about the cardinality of the point cloud M_n, and in what follows, we drop the index n without loss of generality. We draw inspiration from class activation map (CAM) methods [64], which we briefly review next. Our method is trained to find the optimal set of parameters $\boldsymbol{\theta}$ that minimize losses:

$$\arg\min_{\boldsymbol{\theta}} \underbrace{\ell_{\text{cam}}(\boldsymbol{\theta})}_{\text{Section 3.1}} + \underbrace{\ell_{\text{us}}(\boldsymbol{\theta})}_{\text{Section 3.2.1}} + \underbrace{\ell_{\text{match}}(\boldsymbol{\theta})}_{\text{Section 3.2.2}}. \quad (1)$$

Our core contributions are the latter two losses, bringing a significant performance gain compared to simply ℓ_{cam}, as we show in Sec. 4. With $\boldsymbol{\theta}$, we generally denote *all* the trainable parameters of our method.

3.1. Background – Class Activation Maps (CAM)

Given H-dimensional point features $\mathbf{f} \in \mathbb{R}^{M \times H}$ computed from an off-the-shelf backbone network [11], class activation maps [57, 58, 64] use a *linear* layer $\mathcal{L}$ with trainable weights $\boldsymbol{\omega} \in \mathbb{R}^{H \times C}$, to predict point-wise class scores $\mathbf{s} \in [0,1]^{M \times C}$ as $\mathbf{s} = \mathcal{L}(\mathbf{f}; \boldsymbol{\omega})$. To learn dense

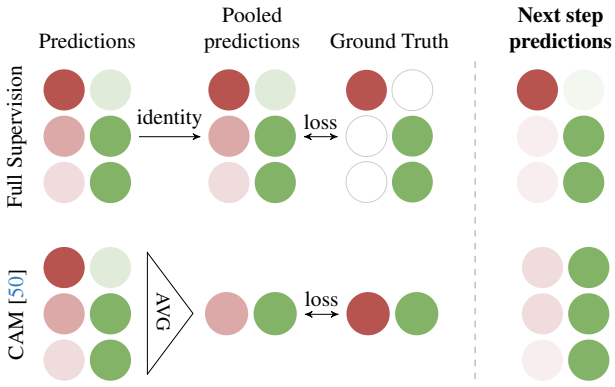


Figure 2. **Failure of CAM [50]** – Given a set of points (vertically) and classifier's scores (red ■ and green ■), CAM firstly averages scores along the dimension of the points and then supervises the average to match scene-level labels. Because of using a single label for all the points, they may be misclassified towards a majority class, even though they correspond to different semantic classes. In contrast, if dense labels are available, the classifier correctly predicts the "red" class for the first point. Note that in our problem, we *do not* assume the availability of dense supervision.

point predictions $\mathbf{y}^{\text{point}} \in \{0,1\}^{M \times C}$ from per-scene class labels $\mathbf{y} \in \{0,1\}^C$, they optimize the loss

$$\ell_{\text{cam}} = \mathcal{H}_C\Big(\underbrace{\frac{1}{M}\sum_{m=1}^{M}\mathbf{s}_m}_{\text{mean score}},\ \mathbf{y}\Big), \quad (2)$$

where $\mathcal{H}_C$ is the sum of binary entropies over C classes:

$$\mathcal{H}_C(\mathbf{s},\mathbf{y}) = \sum_{c=1}^{C}\mathcal{H}(\bar{\mathbf{s}}_c,\ y_c), \quad \bar{\mathbf{s}}_c = \frac{1}{M}\sum_{m=1}^{M}\mathbf{s}_{m,c}. \quad (3)$$

which is different from the mean entropy objective employed by *dense* (fully-supervised) methods:

$$\ell_{\text{dense}} = \frac{1}{M}\sum_{m=1}^{M}\mathcal{H}(\mathbf{s},\ \mathbf{y}^{\text{point}}_m), \quad \forall_{m\in M}\|\mathbf{y}^{\text{point}}_m\|{=}1 \quad (4)$$

where $\mathbf{y}^{\text{point}}_m$ is a one-hot label vector, and $\mathcal{H}$ is the simple cross entropy function. In contrast to Eq. (2), this requires point-wise labels $\mathbf{y}^{\text{point}}$, and uses averaging pooling after the entropy function. Therefore, CAM bypasses the need for the dense label $\mathbf{y}^{\text{point}}$ by moving the global average pooling (GAP) *inside* the entropy calculation. This simplifies its supervision, as only $\mathbf{y}$ is needed at training time, instead of the much richer $\mathbf{y}^{\text{point}}$ label. However, it comes at the cost of generating many false positives at inference time. We demonstrate this effect on a toy example in Figure 2.

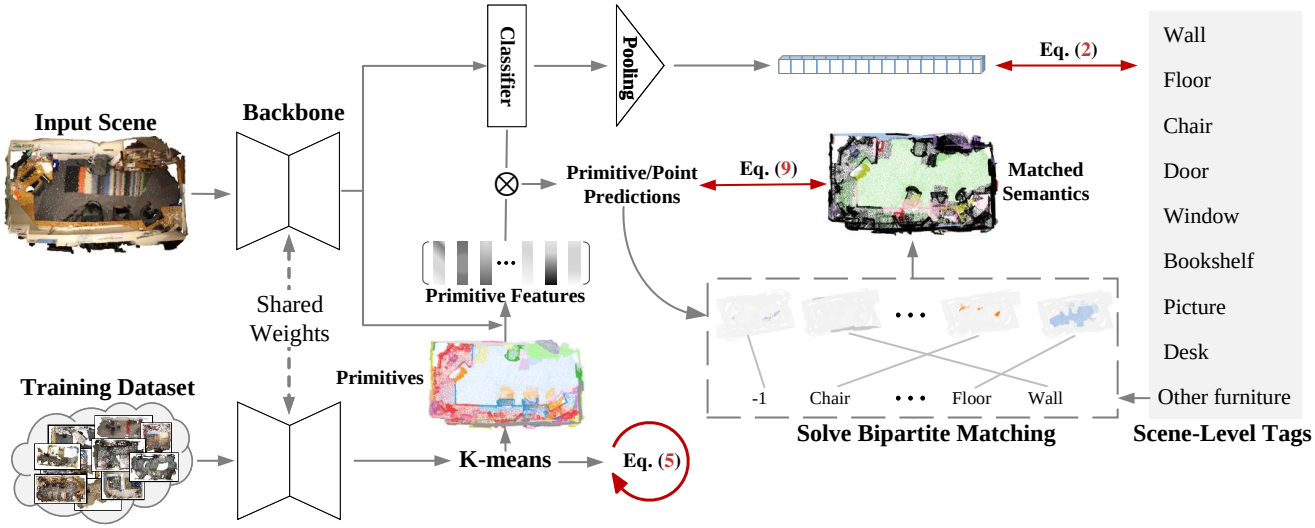


Figure 3. **Framework** – We show the overview of our approach for weakly supervised point cloud segmentation. We leverage K-means performed on features from a point cloud backbone. We then apply bipartite matching to assign semantic labels conservatively to the found clusters, or *primitives*. Finally, we propagate the assigned labels to points as a pseudo-ground truth.

3.2. Dense supervision from scene-level labels

To solve the issue of generating false positives in CAM, we propose a scene-level label "densification" module that maps the scene-level labels $\mathbf{y} \in \mathbb{R}^C$ to the pseudo-point-wise label $\mathbf{y}^{\text{point}} \in \mathbb{R}^{M \times C}$. Such labels allow us to apply dense supervision as in Eq. (4) *without* having access to dense annotations. We achieve this by first learning clusters of features (Sec. 3.2.1), and then devising a way to map learned cluster features to the semantic classes (Sec. 3.2.2).

3.2.1 Unsupervised feature clustering

At the training stage, we collect point features $\{\mathbf{f}_n\}$ from all N point clouds $\mathcal{P}$ and apply K-means clustering to obtain $\mathbf{f}^{\text{prim}} \in \mathbb{R}^{K \times H}$ feature *centroids* or *primitives*[1]. While a single primitive should describe a single object class, this would require our backbone to have built a manifold composed of convex subspaces, which can hardly be expected. Therefore, we use $K \gg C$ to over-segment the feature space.

Primitive warm-up. Note that at the beginning of training features are random, and therefore K-means clusters semantically different features into the same group. To avoid this, we refer [62] to concatenate handcrafted features [38] and color to $\mathbf{f}$ and guide clustering via:

$$\ell_{\text{us}}(\boldsymbol{\theta}) = \mathcal{H}\Big(\text{softmax}\big(\mathbf{f} \cdot (\mathbf{f}^{\text{prim}})^\top\big), \mathbf{A}^{\text{prim}}\Big), \qquad (5)$$

where $\mathbf{A}^{\text{prim}} \in \{0,1\}^{M \times K}$ is an assignment matrix induced by K-means, where each row is a one-hot vector assigning the m-th point to the k-th cluster, and softmax$(\cdot)$ normalizes rows so that they sum to one. In short, this term ties points' and primitives' features so that their inner product reproduces the assignment matrix. Details about these handcrafted features can be found in the `Supplementary`.

[1]Note that we only perform this every 10 epochs as executing K-means clustering on the whole dataset is computationally intensive.

3.2.2 Semantic assignment

The assignment matrix $\mathbf{A}^{\text{prim}}$ does not have a semantic meaning, as the clusters merely collect similar features together. To associate them with scene labels, one can think of various ways. We first discuss the naïve linear assignment and its pitfalls, then describe our bipartite matching strategy.

Naïve assignment. As a simple way to link between clusters and scene labels could be to use a linear classifier $\mathcal{L}(\cdot;\boldsymbol{\omega})$ to classify the primitives. However, as shown in Fig. 5, the linear classifier tends to predict noisy semantic assignment, as it is weakly supervised by ℓ_{cam}.

Bipartite assignment. Instead, we propose to assign semantic labels by optimizing for a bipartite matching between primitives and semantic classes via the Hungarian algorithm [20]:

$$\pi(k) = \underset{c \in [1,C]}{\arg\max} \left(\sum_{k=1}^{K} \mathbf{E}_{k,c} \right), \qquad (6)$$

where $\pi(\cdot) \in \{-1, 0, \ldots, C-1\}$ is now a permutation mapping k-th centroid to one of the C classes. $\mathbf{E} \in \mathbb{R}^{K \times C}$ is a

cost matrix with entries:

$$\mathbf{E}_{k,c} = \mathbf{y}_c \cdot (\bar{\mathbf{f}}_k^{\text{prim}} \cdot \boldsymbol{\omega}_{(\cdot,c)}),\ \bar{\mathbf{f}}_k^{\text{prim}} = \frac{\sum_m \mathbf{A}_{m,k}^{\text{prim}} \odot \mathbf{f}_{m,k}}{\sum_m \mathbf{A}_{m,k}^{\text{prim}}}, \quad (7)$$

where the scene-level label $\mathbf{y}_c$ serves as a prior to avoid assigning a non-existing class to a cluster. We use a special index -1 for $\pi(\cdot)$ for all unassigned primitives, which we ignore during loss computation. With this assignment, we then obtain per-primitive pseudo ground truth as:

$$\mathbf{y}^{\text{prim}} = \{\mathbf{y}_k^{\text{prim}}\}, \quad \mathbf{y}_k^{\text{prim}} = \mathcal{L}(\bar{\mathbf{f}}_{\pi(k)}^{\text{prim}}; \boldsymbol{\omega}). \quad (8)$$

Then, we supervise primitives to correspond to the assigned pseudo labels:

$$\ell_{\text{match}} = \mathcal{H}(\mathcal{L}(\bar{\mathbf{f}}^{\text{prim}}; \boldsymbol{\omega}),\ \mathbf{y}^{\text{prim}}). \quad (9)$$

Note here that while we only supervise $\bar{\mathbf{f}}^{\text{prim}}$, this supervision naturally propagates to each point when calculating $\bar{\mathbf{f}}^{\text{prim}}$ from $\mathbf{f}$ as in Eq. (7).

4. Experiments

Following the baselines, we benchmark our method on ScanNet [9] and S3DIS [2] datasets for large-scale point cloud semantic segmentation. ScanNet [9] consists of 1,201 training scenes, 312 validation scenes, and 100 test scenes, with 20 semantic classes. S3DIS [2] contains 272 rooms from 6 indoor areas and has 13 semantic categories. We compare our approach to the existing scene-level supervised methods including PCAM [50], MPRM [50], MIL-Seg [36], WYPR [36], MIL-Trans [57], and the recent state-of-the-art MIT [58]. We also provide the results for fully supervised approaches ScanNet [9] and PointNet++ [35]. Following the experimental setting in the baselines [35, 50, 54, 57], we select `Area 5` as the test scene and use other areas for training. We extract scene-level labels from the provided dense ground-truth labels. We use mean Intersection over Union (mIoU) as the metric to evaluate the semantic segmentation results.

4.1. Implementation Details

We use SparseConv [11] as our point cloud backbone. The implementation of CAM baseline is provided by Ren et al. [36]. The number of epochs is 400. We set K=300 and perform K-means every 10 epochs. As performing K-means on all the points is computationally prohibitive, we cluster points to super-voxels, following the same algorithm as in [62]. Our warm-up procedure lasts for 250 training epochs. We train our model on NVIDIA 3090 with batch size 6, and use the AdamW [29] optimizer with OneCycleLR [41] policy and learning rate set to 0.01.

	Methods	ScanNet [9]			S3DIS [2]
		Train	Val	Test	Test
Full	ScanNet [9]	-	-	30.6	-
	PointNet++ [35]	-	-	33.9	-
Scene-level	PCAM [50]⋆	22.1	-	-	-
	MPRM [50]⋆	24.4	-	-	10.3
	MIL-Seg [36]	-	20.7	-	-
	WYPR [36]	-	29.6	24.0	22.3
	MIL-Trans[57]	-	26.2	-	12.9
	MIT [58]⋆	-	31.6	26.4	23.1
	Ours	-	<u>33.7</u>	<u>30.9</u>	<u>23.6</u>
	Ours⋆	-	**38.1**	**35.1**	**25.7**

Table 1. **Quantitative Results** – We show the results on the ScanNet [9] and S3DIS [2] datasets. Our method achieves state-of-art performance compared to other scene-level weakly supervised methods ("Scene-level"). We further compare with the *fully supervised* methods ("Full"). After the bootstrapping (denoted as ⋆), our method matches the performance of fully supervised methods.

Bootstrapping. We follow the scene-level supervised methods to further boost the performance via bootstrapping [50, 58]. Specifically, we re-train the segmentation network with semantic prediction results after the initial convergence. As in Yang et al. [58], the network is a sparse residual U-Net [52]. To mitigate the noise in the initial prediction, we supervise the network with the subset of points with semantic predictions of high quality. Different from existing works that rely on complex filtering steps [50, 58], we simply keep the points of semantic prediction consistent with the scene-level label. We note that our method outperforms all the baselines by a large margin *even without bootstrapping*.

4.2. Quantitative Results – Tab. 1

We compare our method with scene-level supervised and fully supervised baselines. Our approach significantly surpasses scene-level baselines—it displays an 8.7 boost in mIoU on ScanNet and a 2.6 boost on S3DIS. Furthermore, we are among the first to outperform two of the fully supervised methods, ScanNet [9] and PointNet++ [35] on the ScanNet [9] dataset. It shows that weakly supervised approaches have a great potential for precise segmentation. We note that our method achieves significantly better results on ScanNet than S3DIS. We suspect that the size of S3DIS is the main culprit—it is five times smaller than ScanNet, degrading K-means performance. Therefore, even if dense labels are not required, our unsupervised primitives still need rich data. We also observe another issue. S3DIS contains *incoherent* ground-truth scene-level labels, further limiting our method; please refer to Section 4.5.

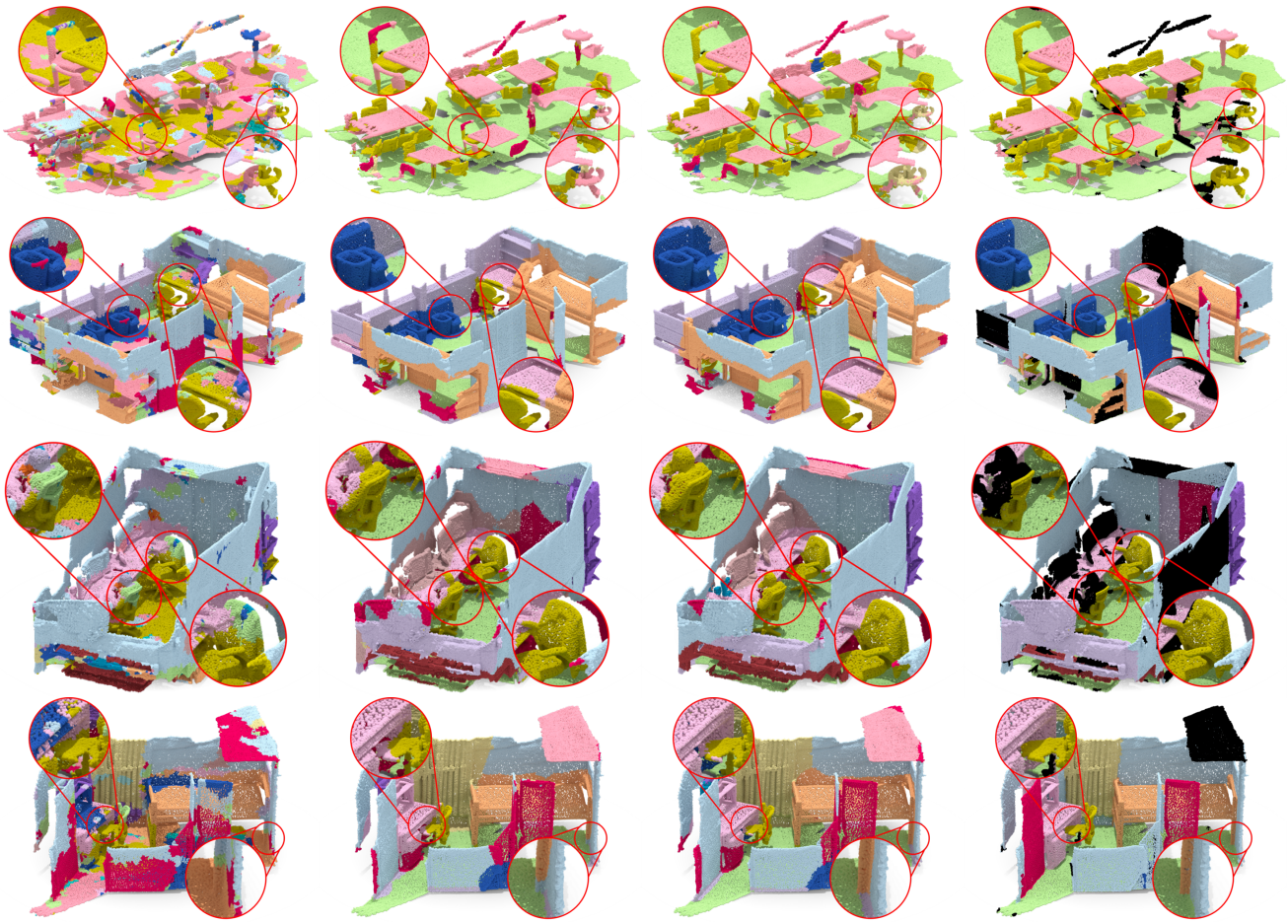


Figure 4. **Qualitative results** – Our method (**Ours**), predicts more accurate and more spatially consistent labels. In contrast, the CAM-based baseline method often produces noisy classes, which is easily visible in the first row of results. Bootstrapping (**Ours***) further improves the performance.

4.3. Qualitative Results – Fig. 4

We compare our approach qualitatively in Fig. 4 with the CAM baseline [50]. Our method performs significantly better even without retraining. It can distinguish semantically similar objects close to each other spatially (the first row in Fig. 4). Our predicted labels are also more spatially smooth than the baseline (zoom-ins), because unsupervised primitives can group close points into coherent segments. Furthermore, the primitive matching removes many erroneous supervision signals, lowering the noise level in our results (last row in Fig. 4).

4.4. Primitive Matching – Fig. 5

We further provide the analysis of our novel primitive matching method. We start by defining a simple baseline, "Naïve matching" which aligns primitives and semantic labels by $\mathcal{L}(\cdot;\boldsymbol{\omega})$. In Figure 5, we visually demonstrate that our proposed bipartite matching is critical to correctly associate scene-level tags to points by comparing matched semantics with ground truth. The classifier $\mathcal{L}(\cdot;\boldsymbol{\omega})$ predicts incorrect semantics for primitives, which naïve matching cannot rule out. To lessen the impact of wrong predictions, we match only one primitive for each category and drop the others (black points in Fig. 5). The primitives guided by naïve matching may also fail in separating different semantic regions (*e.g.*, table and chair in the second row), indicating that our matching mechanism also helps in unsupervised clustering. Furthermore, some semantic tags may not correspond to any primitives in naïve matching due to the limitation of $\mathcal{L}(\cdot;\boldsymbol{\omega})$. For example, in the third row, naïve matching fails to find corresponding primitives for "window" and "otherfurniture". In contrast, our method ensures that every scene-level label has a proper corresponding primitive.

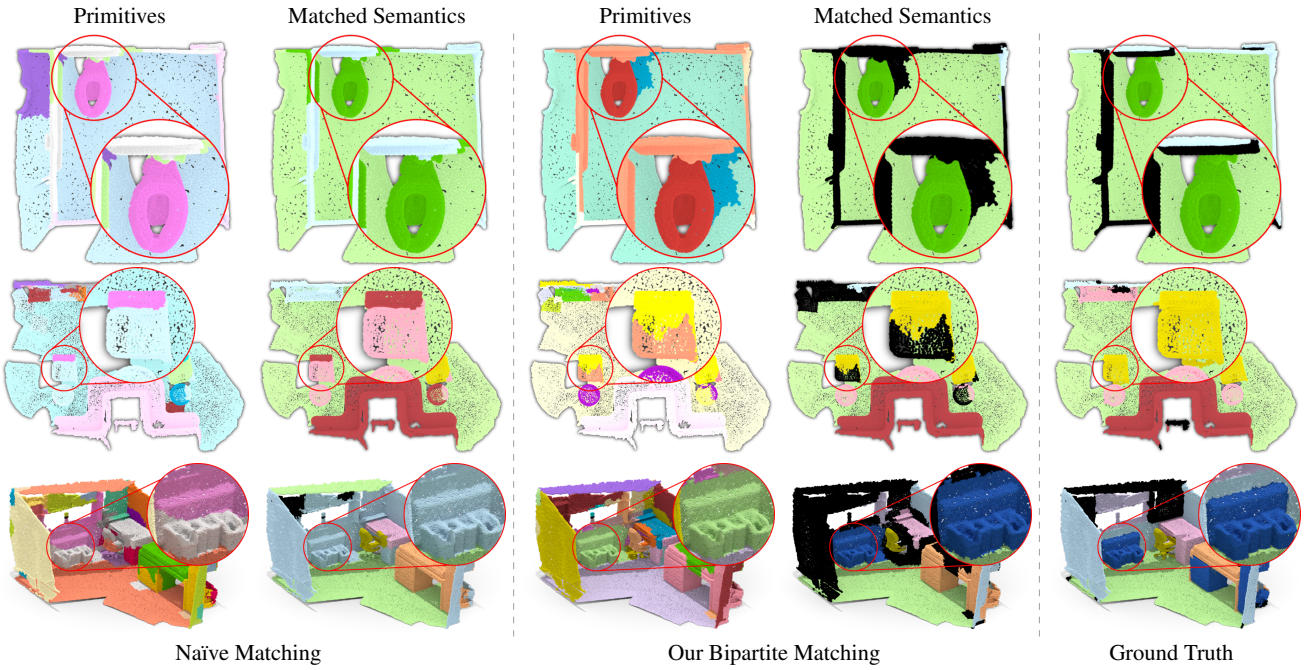


Figure 5. **Primitive Matching** – We show that our proposed bipartite matching (Eq. (6)) aligns *primitives* better to semantic labels than naïve matching. We color primitives randomly and matched semantics according to the class label. Black colors denote unmatched primitives or ignored points. We compare the quality to the ground truth provided in the dataset.

Method	Initial	Reduced
CAM	8.4	12.6 ± 0.6
Ours	23.6	30.5 ± 0.9

Table 2. **Reduce Label Co-occurrence** – In the table, "Initial" refers to the original scene-level annotations, while "Reduced" refers to randomly removing "floor" from one scene and "ceiling" from another. We report the average mIoU of five random removal tests.

4.5. Label Co-occurrence in S3DIS – Tab. 2

To address the discrepancy phenomenon between ScanNet and S3DIS results in Tab. 1, we analyze the scene-level annotations on the S3DIS dataset. We find that the "floor", "ceiling" and "wall" categories appear in all 204 training scenes (refer to the `Supplementary`). It implies that when calculating semantic-related losses from Eqs. (2) and (9), mislabeling those categories interchangeably (*e.g.*, floor to ceiling) does not affect the loss value. Therefore, this semantic mapping is ill-posed, leading to an inferior and unstable training of the model.

Logically, we can resolve the problem by removing at least two of those categories. In this experiment, we explore removing "floor" and "ceiling" which are mostly planar surfaces and can be easily discarded in most cases. However, "wall" is more challenging to remove as it may contain other categories like "windows" or "boards". Thus, we randomly select two scenes from S3DIS and remove "floor" in one of the scenes and "ceiling" in another. We present our findings in Tab. 2, with CAM serving as a baseline. We show that removing co-occurring labels in the dataset can notably improve the performance of both methods. For our approach, the difference reaches a 6.9 increase in mIoU. This experiment underscores the importance of label diversity and highlights the advantages of our proposed approach. We leave further analysis of the label diversity in scene-level weakly supervised learning as future work.

4.6. Ablation Studies

We perform ablation studies on ScanNet, which is more stable than S3DIS under scene-level setting as analyzed in Sec. 4.5. We show that training the primitives in tandem with CAM is essential. Then, we examine the importance of ℓ_{us} and ℓ_{match}. Next, we search for the optimal K value in primitive learning. Finally, we analyze the primitive-based inference regime.

Training Primitives – Tab. 3. Leveraging pre-trained primitives in the CAM-based methods is a logical idea as primitives can serve as regularizers and improve performance directly. We implement this idea by applying the affinity matrix $\mathbf{A}^{\text{prim}}$ to supervise the method trained on the

Primitive usage strategy	mIoU
Pretrained primitives [62]	25.4
CAM	19.2
CAM + pretrained primitives	23.7
Ours	**33.7**

Table 3. **Training primitives** – We highlight the importance of joint training primitives with CAM. We compare our approach by using pre-trained primitives [62], CAM alone, and a simple combination of both.

Loss	mIoU	Δ
ℓ_{cam}	19.2	0.0
ℓ_{cam}+ ℓ_{us}	24.9	5.9
ℓ_{cam}+ ℓ_{match}	16.9	-2.3
ℓ_{cam}+ ℓ_{us}+ $\ell^{\star}_{\text{match}}$	20.5	1.3
ℓ_{cam}+ ℓ_{us}+ ℓ_{match}	**33.7**	**14.5**

Table 4. **Our objective** – We show how ℓ_{cam}, ℓ_{us} and ℓ_{match} in Eq. (1) affect the final performance. $\ell^{\star}_{\text{match}}$ refers to the naïve matching loss.

CAM loss Eq. (2) only. As shown in in Tab. 3, the supervision improves the CAM performance marginally. However, it is still lower than primitive clustering from [62]. This suggests that simply integrating trained primitives into scene-level weakly supervised learning framework cannot fully realize the potential of both primitives and CAM. Our full design trains the primitives in tandem with CAM performs best.

Loss components – Tab. 4. We investigate the effect ℓ_{us} and ℓ_{match} have on our method. We show the results in Tab. 4 and use ℓ_{cam} as a baseline. Adding ℓ_{us} to the objective leads to an mIoU increase by 5.9. We argue that ℓ_{us} helps to maintain primitive features and features of the corresponding points close to each other, *i.e.*, they do not diverge throughout the training. When using ℓ_{match} alone, the performance decreases by 2.3. However, using *both* ℓ_{us} and ℓ_{match} gives the best results. We suggest that without ℓ_{us}, ℓ_{match} cannot benefit from the bipartite matching as the centroid features diverge from the point features over training time. This effect, however, does not apply when applying a naïve matching (denotes as $\ell^{\star}_{\text{match}}$).

Number of primitives – Tab. 5. The number of primitives K affects the performance of the proposed framework. At low values K=100, the number of primitives is insufficient to fully capture the semantically different parts of objects in the dataset, resulting in the lowest mIoU as shown in Tab. 5. On the other hand, the mIoU saturates quickly as K increases. We achieve the best performance at K=300, which is also suggested in [62]. When K becomes larger, the performance decreases slightly at the cost of increased training time. More primitives result in clustering smaller regions, which may be more vulnerable to noisy labels [59].

K	100	200	300	400	500
mIoU	24.3	32.7	**33.7**	33.6	32.9

Table 5. **Number of primitives** – We compare the results between different number of primitives K. We achieve the best performance with K=300.

Label prediction	mIoU	Δ
With primitives only	32.5	0.0
With our classifier	**33.7**	**1.2**

Table 6. **Semantic segmentation via primitives** – We show that our primitives can already effectively segment points alone (the first row). We achieve our state-of-the-art results by using our trained classifier.

Semantic segmentation via primitives – Tab. 6. We further show the ability of our primitives to serve as a classifier. We simply classify the points into primitives using the primitive feature $\mathbf{f}^{\text{prim}}$ and then propagate the semantic labels from primitives to points, thereby resulting in the semantic segmentation of input points. Surprisingly, this simple primitive-based method achieves semantic segmentation performance on par with the trained classifier $\mathcal{L}$ in our full model, highlighting the effectiveness of our primitive learning.

5. Conclusions

We have presented a new method for weakly supervised point cloud semantic segmentation which uses unsupervised primitives to conservatively propagate scene-level annotations to points by bipartite matching. Our approach achieves new state-of-the-art weakly-supervised point cloud segmentation and performs competitively with fully supervised methods while being much more label efficient. Additionally, we have demonstrated a significant issue in the S3DIS dataset [2], which impacts extensively this research field. We have shown how to solve it and how it impacts our method's performance. We hope our work will inspire future weakly supervised point cloud segmentation research.

Our approach is an important milestone towards democratizing low-cost, high-quality point cloud segmentation. We expect that in the near future, we will see the rise of weakly supervised methods that, by means of large language models [53], can achieve performance greater than that of fully supervised approaches.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 42201481, the Scientific Research Foundation of Hunan Education Department under Grant 21B0332, in part by the Science and Technology Plan Project Fund of Hunan Province under Grant 2023JJ40024. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the A100 GPU used for this research. Lastly, we would like to thank Yuhe Jin for the insightful discussions.

References

[1] Rolf Adams and Leanne Bischof. Seeded Region Growing. *PAMI*, 1994. 1

[2] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D Semantic Parsing of Large-Scale Indoor Spaces. In *CVPR*, 2016. 2, 5, 8, 1

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In *ECCV*, 2020. 3

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *NeurIPS*, 2020. 3

[5] Mingmei Cheng, Le Hui, Jin Xie, and Jian Yang. SSPC-Net: Semi-supervised Semantic 3D Point Cloud Segmentation Network. In *AAAI*, 2021. 2

[6] Julian Chibane, Francis Engelmann, Tuan Anh Tran, and Gerard Pons-Moll. Box2Mask: Weakly Supervised 3D Semantic Instance Segmentation Using Bounding Boxes. In *ECCV*, 2022. 2

[7] Jang Hyun Cho, Utkarsh Mall, Kavita Bala, and Bharath Hariharan. PiCIE: Unsupervised Semantic Segmentation using Invariance and Equivariance in Clustering. In *CVPR*, 2021. 2, 3

[8] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *CVPR*, 2019. 1

[9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017. 1, 2, 5

[10] Huan-ang Gao, Beiwen Tian, Pengfei Li, Hao Zhao, and Guyue Zhou. DQS3D: Densely-matched Quantization-aware Semi-supervised 3D Detection. In *ICCV*, 2023. 2

[11] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In *CVPR*, 2018. 3, 5

[12] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep Learning for 3D Point Clouds: A Survey. *PAMI*, 2020. 1

[13] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring Data-Efficient 3D Scene Understanding with Contrastive Scene Contexts. In *CVPR*, 2021. 2

[14] Qingyong Hu, Bo Yang, Guangchi Fang, Yulan Guo, Ales Leonardis, Niki Trigoni, and Andrew Markham. Sqn: Weakly-supervised semantic segmentation of large-scale 3d point clouds. In *ECCV*, 2022. 1, 2

[15] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. SensatUrban: Learning Semantics from Urban-Scale Photogrammetric Point Clouds. *IJCV*, 2022. 1

[16] Li Jiang, Shaoshuai Shi, Zhuotao Tian, Xin Lai, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Guided Point Contrastive Learning for Semi-supervised Point Cloud Semantic Segmentation. In *ICCV*, 2021. 2

[17] Yuhe Jin, Weiwei Sun, Jan Hosang, Eduard Trulls, and Kwang Moo Yi. TUSK: Task-Agnostic Unsupervised Keypoints. In *NuerIPS*, 2022. 2, 3

[18] Sanghyun Jo, In-Jae Yu, and Kyungsu Kim. MARS: Model-agnostic Biased Object Removal without Additional Supervision for Weakly-Supervised Semantic Segmentation. In *ICCV*, 2023. 3

[19] Lingdong Kong, Jiawei Ren, Liang Pan, and Ziwei Liu. LaserMix for Semi-Supervised LiDAR Semantic Segmentation. In *CVPR*, 2023. 2

[20] Harold W Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955. 2, 3, 4

[21] Mengtian Li, Yuan Xie, Yunhang Shen, Bo Ke, Ruizhi Qiao, Bo Ren, Shaohui Lin, and Lizhuang Ma. HybridCR: Weakly-Supervised 3D Point Cloud Semantic Segmentation via Hybrid Contrastive Regularization. In *CVPR*, 2022. 2

[22] Kangcheng Liu, Yuzhi Zhao, Qiang Nie, Zhi Gao, and Ben M Chen. Weakly Supervised 3D Scene Segmentation with Region-Level Boundary Awareness and Instance Discrimination. In *ECCV*, 2022. 2

[23] Leyao Liu, Tao Kong, Minzhao Zhu, Jiashuo Fan, and Lu Fang. ClickSeg: 3D Instance Segmentation with Click-Level Weak Annotations. *arXiv*, 2023. 2

[24] Lizhao Liu, Zhuangwei Zhuang, Shangxin Huang, Xunlong Xiao, Tianhang Xiang, Cen Chen, Jingdong Wang, and Mingkui Tan. CPCM: Contextual Point Cloud Modeling for Weakly-supervised Point Cloud Semantic Segmentation. In *ICCV*, 2023. 2

[25] Minghua Liu, Yin Zhou, Charles R. Qi, Boqing Gong, Hao Su, and Dragomir Anguelov. LESS: Label-Efficient Semantic Segmentation for LiDAR Point Clouds. In *ECCV*, 2022. 1, 2

[26] Zhengzhe Liu, Xiaojuan Qi, and Chi-Wing Fu. One Thing One Click: A Self-Training Approach for Weakly Supervised 3D Semantic Segmentation. In *CVPR*, 2021. 2

[27] Stuart Lloyd. Least squares quantization in PCM. *IEEE transactions on information theory*, 1982. 2

[28] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-Centric Learning with Slot Attention. *NeurIPS*, 2020. 3

[29] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019. 5

[30] Alexey Nekrasov, Jonas Schult, Or Litany, Bastian Leibe, and Francis Engelmann. Mix3D: Out-of-Context Data Augmentation for 3D Scenes. In *3DV*, 2021. 1

[31] Yinyu Nie, Ji Hou, Xiaoguang Han, and Matthias Nießner. RfD-Net: Point Scene Understanding by Semantic Instance Reconstruction. In *CVPR*, 2021. 1

[32] Dim P Papadopoulos, Alasdair DF Clarke, Frank Keller, and Vittorio Ferrari. Training Object Class Detectors from Eye Tracking Data. In *ECCV*, 2014. 2

[33] Jeremie Papon, Alexey Abramov, Markus Schoeler, and Florentin Worgotter. Voxel Cloud Connectivity Segmentation-supervoxels for Point Clouds. In *CVPR*, 2013. 1

[34] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. OpenScene: 3D Scene Understanding with Open Vocabularies. In *CVPR*, 2023. 1

[35] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NIPS*, 2017. 1, 5

[36] Zhongzheng Ren, Ishan Misra, Alexander G Schwing, and Rohit Girdhar. 3D Spatial Recognition without Spatially Labeled 3D. In *CVPR*, 2021. 1, 2, 5

[37] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *ICRA*, 2011. 1

[38] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *IROS*, 2008. 4, 1

[39] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning Feature Matching with Graph Neural Networks. In *CVPR*, 2020. 3

[40] Jonathan Sauder and Bjarne Sievers. Self-Supervised Deep Learning on Point Clouds by Reconstructing Space. In *NeurIPS*, 2019. 2

[41] Leslie N. Smith and Nicholay Topin. Super-Convergence: Very Fast Training of Residual Networks Using Large Learning Rates. *arXiv*, 2018. 5

[42] Russell Stewart, Mykhaylo Andriluka, and Andrew Y Ng. End-to-end People Detection in Crowded Scenes. In *CVPR*, 2016. 3

[43] Yongyi Su, Xun Xu, and Kui Jia. Weakly Supervised 3D Point Cloud Segmentation via Multi-Prototype Learning. *CSVT*, 2023. 2

[44] Weiwei Sun, Andrea Tagliasacchi, Boyang Deng, Sara Sabour, Soroosh Yazdani, Geoffrey Hinton, and Kwang Moo Yi. Canonical Capsules: Self-Supervised Capsules in Canonical Pose. In *NeurIPS*, 2021. 2

[45] Weiwei Sun, Daniel Rebain, Renjie Liao, Vladimir Tankovich, Soroosh Yazdani, Kwang Moo Yi, and Andrea Tagliasacchi. NeuralBF: Neural Bilateral Filtering for Top-down Instance Segmentation on Point Clouds. In *WACV*, 2023. 1

[46] An Tao, Yueqi Duan, Yi Wei, Jiwen Lu, and Jie Zhou. SegGroup: Seg-Level Supervision for 3D Instance and Semantic Segmentation. *TIP*, 2022. 1, 2

[47] Ozan Unal, Dengxin Dai, and Luc Van Gool. Scribble-Supervised LiDAR Semantic Segmentation. In *CVPR*, 2022. 1, 2

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NIPS*, 2017. 2

[49] Peng-Shuai Wang. OctFormer: Octree-based Transformers for 3D Point Clouds. *TOG*, 2023. 1

[50] 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 *CVPR*, 2020. 1, 2, 3, 5, 6

[51] Linshan Wu, Leyuan Fang, Xingxin He, Min He, Jiayi Ma, and Zhun Zhong. Querying Labeled for Unlabeled: Cross-Image Semantic Consistency Guided Semi-Supervised Semantic Segmentation. *PAMI*, 2023. 3

[52] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding. In *ECCV*, 2020. 2, 5

[53] Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. PointLLM: Empowering Large Language Models to Understand Point Clouds. *arXiv*, 2023. 8

[54] Xun Xu and Gim Hee Lee. Weakly Supervised Semantic Point Cloud Segmentation: Towards 10× Fewer Labels. In *CVPR*, 2020. 1, 2, 5

[55] Xiuwei Xu, Yifan Wang, Yu Zheng, Yongming Rao, Jie Zhou, and Jiwen Lu. Back to Reality: Weakly-supervised 3D Object Detection with Shape-guided Label Enhancement. In *CVPR*, 2022. 2

[56] Zongyi Xu, Bo Yuan, Shanshan Zhao, Qianni Zhang, and Xinbo Gao. Hierarchical Point-based Active Learning for Semi-supervised Point Cloud Semantic Segmentation. In *ICCV*, 2023. 2

[57] Cheng-Kun Yang, Ji-Jia Wu, Kai-Syun Chen, Yung-Yu Chuang, and Yen-Yu Lin. An MIL-Derived Transformer for Weakly Supervised Point Cloud Segmentation. In *CVPR*, 2022. 2, 3, 5

[58] Cheng-Kun Yang, Min-Hung Chen, Yung-Yu Chuang, and Yen-Yu Lin. 2D-3D Interlaced Transformer for Point Cloud Segmentation with Scene-Level Supervision. In *ICCV*, 2023. 1, 2, 3, 5

[59] Shuquan Ye, Dongdong Chen, Songfang Han, and Jing Liao. Learning with Noisy Labels for Robust Point Cloud Segmentation. In *ICCV*, 2021. 8

[60] Yachao Zhang, Zonghao Li, Yuan Xie, Yanyun Qu, Cuihua Li, and Tao Mei. Weakly Supervised Semantic Segmentation for Large-Scale Point Cloud. In *AAAI*, 2021. 2

[61] Yachao Zhang, Yanyun Qu, Yuan Xie, Zonghao Li, Shanshan Zheng, and Cuihua Li. Perturbed Self-Distillation: Weakly Supervised Large-Scale Point Cloud Semantic Segmentation. In *ICCV*, 2021. 2

[62] Zihui Zhang, Bo Yang, Bing Wang, and Bo Li. GrowSP: Unsupervised Semantic Segmentation of 3D Point Clouds. In *CVPR*, 2023. 2, 4, 5, 8, 1

[63] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. SESS: Self-Ensembling Semi-Supervised 3D Object Detection. In *CVPR*, 2020. 2

[64] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning Deep Features for Discriminative Localization. In *CVPR*, 2016. 2, 3

[65] Tianfei Zhou, Meijie Zhang, Fang Zhao, and Jianwu Li. Regional Semantic Contrast and Aggregation for Weakly Supervised Semantic Segmentation. In *CVPR*, 2022. 3

Densify Your Labels: Unsupervised Clustering with Bipartite Matching for Weakly Supervised Point Cloud Segmentation

Supplementary Material

We provide additional implementation details in Appendix A, more qualitative and quantitative results in Appendix B, a description of the label co-occurrence issue in Appendix C, and analysis of CAM's limitations in Appendix D. Please also refer to densify-your-labels.github.io for the interactive visualizations of our results.

A. Additional implementation details

A.1. Supervoxel construction in K-means clustering

We construct supervoxels to accelerate K-means clustering as presented in Zhang et al. [62]. Specifically, we combine VCCS [33] and region growing [1] to construct supervoxels for all the scenes. If a growing region covers a VCCS patch, we merge the VCCS patch into the former. We implement these methods based on Point Cloud Library [37] and use the same parameter settings as in Zhang et al. [62]. Note that the supervoxel construction is necessary only during training.

A.2. Handcrafted features during warm-up

In Sec. 3.2.1, we mention regularizing the primitive clustering with handcrafted features at the beginning of the training. Specifically, we follow Zhang et al. [62] to compute the average RGB and PFH [38] features for each supervoxel, and then concatenate them with features from the backbone, forming the input feature of K-means.

A.3. No matching loss at the beginning of training

We further drop the ℓ_{match} from Eq. (9) during the first 60 training epochs. Our matching objective relies on the quality of the trained classifier $\mathcal{L}$ whereas, at the beginning of training, the prediction of $\mathcal{L}$ is noisy. We observe that this causes the model to get stuck with the incorrect assignment and harms the final performance.

A.4. Semantic assignment filtering

We observe that semantically similar primitives are potentially assigned with two different labels, which harms performance. To address this issue, we re-apply K-means to cluster the initial 300 primitives into 120 groups after the warm-up stage. When a new group has more than two primitives with different assigned labels, we un-assign all the primitives of that group by setting their labels to a special index -1. Otherwise, we propagate a label to other primitives inside the new cluster. We notice a slight mIoU improvement when using this procedure. Note that our method outperforms baselines by a large margin even without it.

B. More results

B.1. Per-class results – Tab. 7 and Tab. 8

We present in Tab. 7 and Tab. 8 extended versions of our quantitative results and show mIoU score for each semantic class existing in the ScanNet [9] and S3DIS [2] datasets. In ScanNet, our method excels in most of the categories, notably in "desk", "table", "chair" and "floor". However, it easily confuses classes related to the "wall" such as "window" or "picture". Similarily, it fails to segment classes such as "beam", "board" and "clutter". We suspect that points of those classes are rare in data, which potentially degrades our unsupervised primitives and also other methods. Nevertheless, our method still performs best among other baselines on average.

B.2. More qualitative results

We show additional segmentation results for the ScanNet scenes in Fig. 6. Please refer to densify-your-labels.github.io for the interactive visualization of point clouds from our experiments.

C. Label Co-occurrence in S3DIS – Tab. 9

To show label co-occurrence in S3DIS dataset, we statistic the frequency of scene-wise semantic labels appearing on the S3DIS training dataset in Tab. 9. The "ceiling","floor" and "wall" appear simultaneously in all the scenes, spending the model's capacity on learning to segment those labels even though they do not bring any information to the task. In Sec. 4.5, we mention randomly removing "ceiling" and "floor" in different scenes, significantly improving the performance. This experiment demonstrates the value of label diversity in point cloud weakly supervised learning. We can avoid those problems while building training datasets or during the label-collecting stage.

D. Analysis of CAM's limitation

We ground the limitations existing in CAM-based methods [50] in a more formal framework. We present the visualization of that framework in Fig. 2.

The CAM-based loss, ℓ_{cam} in Eq. (2), has the one-to-many mapping between points and semantic labels—the model is trained to assign multiple labels to a single point.

Table 7. **Per-class semantic segmentation results on ScanNet** – Our method outperforms the existing state-of-the-art methods [36, 50, 57, 58] by a significant margin. "sc" refers to the "shower curtain". "⋆" indicates the bootstrapping strategy.

Method	eval.	wall	floor	cabinet	bed	chair	sofa	table	door	window	shelf	picture	counter	desk	curtain	fridge	sc	toilet	sink	bathtub	other	IoU
PCAM⋆ [50]	train	54.9	48.3	14.1	34.7	32.9	45.3	26.1	0.6	3.3	46.5	0.6	6.0	7.4	26.9	0.0	6.1	22.3	8.2	52.0	6.1	22.1
MPRM⋆ [50]	train	47.3	41.1	10.4	43.2	25.2	43.1	21.5	9.8	12.3	45.0	9.0	13.9	21.1	40.9	1.8	29.4	14.3	9.2	39.9	10.0	24.4
MIL-seg [36]	val	36.4	36.1	**13.5**	37.9	25.1	31.4	9.6	**18.3**	19.8	33.1	**7.9**	20.3	21.7	32.5	6.4	14.0	7.9	14.7	19.4	8.5	20.7
WYPR [36]	val	58.1	33.9	5.6	56.6	29.1	45.5	19.3	15.2	**34.2**	33.7	6.8	**33.3**	22.1	**65.6**	6.6	36.3	18.6	**24.5**	39.8	6.6	29.6
MIL-Trans [57]	val	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	26.2
MIT⋆ [58]	val	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	31.1
Ours	val	53.5	84.6	11.6	55.5	48.1	56.3	35.2	10.8	17.3	62.3	0.9	4.7	33.0	43.1	6.3	24.7	50.6	13.4	46.8	14.7	33.7
Ours⋆	val	**58.7**	**87.3**	13.3	**60.2**	**52.3**	**61.9**	**36.2**	12.3	22.4	**72.1**	1.5	5.7	**36.4**	53.3	**7.3**	**38.6**	**61.0**	12.1	**53.5**	**16.2**	**38.1**

Table 8. **Per-class semantic segmentation results on S3DIS** – Our method performs best in terms of the average mIoU. "⋆" indicates the bootstrapping strategy. Please note that the existing state-of-the-art methods [36, 50, 57, 58] do not report per-class evaluations.

Method	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter	mIoU
MPRM⋆ [50]	-	-	-	-	-	-	-	-	-	-	-	-	-	10.3
WYPR [36]	-	-	-	-	-	-	-	-	-	-	-	-	-	22.3
MIL-Trans [57]	-	-	-	-	-	-	-	-	-	-	-	-	-	12.9
MIT⋆ [58]	-	-	-	-	-	-	-	-	-	-	-	-	-	23.1
Ours	75.5	82.0	47.1	0.0	**2.3**	**2.9**	**8.0**	21.0	21.0	4.0	42.3	0.0	**1.0**	23.6
Ours⋆	**79.1**	**86.6**	**51.8**	0.0	0.3	0.5	7.6	**30.6**	**26.4**	**5.6**	**45.5**	0.0	0.7	**25.7**

Table 9. **Label co-occurrence matrix** – We show that "ceiling","floor" and "wall" appear in all scenes in the S3DIS [2] training set. Using those labels hinders the performance of our method.

	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
ceiling	**204**	**204**	**204**	104	86	80	190	138	141	22	126	77	202
floor	**204**	**204**	**204**	104	86	80	190	138	141	22	126	77	202
wall	**204**	**204**	**204**	104	86	80	190	138	141	22	126	77	202
beam	104	104	104	104	60	65	97	91	87	11	81	59	104
column	86	86	86	60	86	63	83	74	77	13	75	49	85
window	80	80	80	65	63	80	79	78	79	13	76	56	80
door	190	190	190	97	83	79	190	133	136	20	124	76	189
table	138	138	138	91	74	78	133	138	129	21	113	73	138
chair	141	141	141	87	77	79	136	129	141	21	112	74	140
sofa	22	22	22	11	13	13	20	21	21	22	15	9	22
bookcase	126	126	126	81	75	76	124	113	112	15	126	67	126
board	77	77	77	59	49	56	76	73	74	9	67	77	77
clutter	202	202	202	104	85	80	189	138	140	22	126	77	202

As a result, it incorrectly maximizes the class score of points to the non-belonging classes.

We present the gradient issue of CAM by calculating the gradient sign of point-wise class scores to different classes where the positive gradient encourages points to be categorized as the class and vice-versa. Specifically, we calculate the point sign of point scores derived from *dense supervision* Eq. 4 as

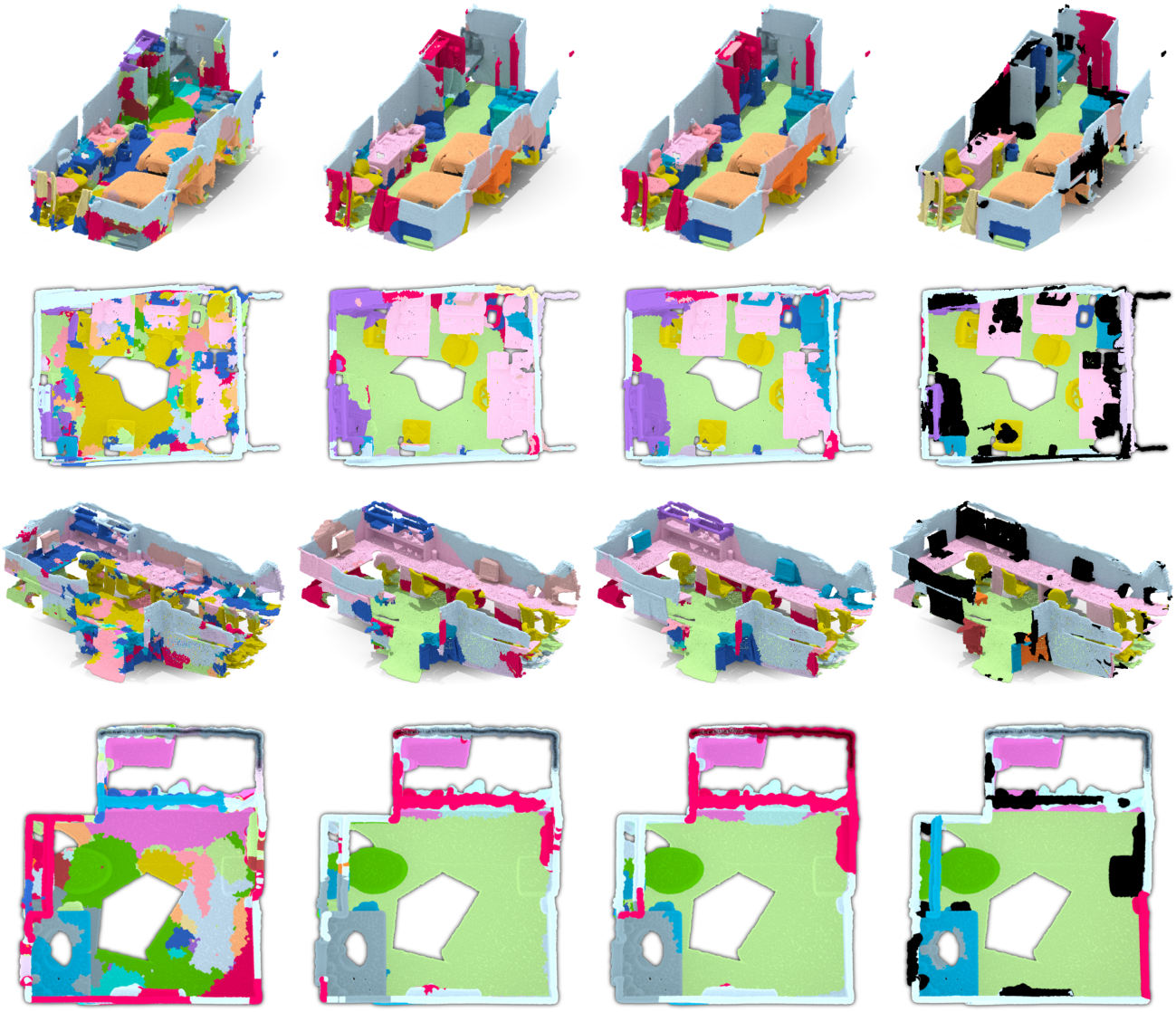
$$\text{sign}\left(\frac{\partial \mathbf{s}_{m,c}}{\partial \mathbf{y}^{\text{point}}_{m,c}}\right) = \begin{cases} +, & \text{if } \mathbf{y}^{\text{point}}_{m,c} = 1 \\ -, & \text{otherwise} \end{cases} \tag{10}$$

where $\mathbf{y}^{\text{point}}_{m,c}$ is the m-th point's label at C-th class. Due to the one-hot label $\mathbf{y}^{\text{point}}$, only one class contributes the positive gradient, contracting m-th point closer to one of the classes.

Similarly, we also calculate the gradient sign of *scene-level supervision* using CAM derived from Eq. 2 and Eq. 3 as

$$\text{sign}\left(\frac{\partial \mathbf{s}_{m,c}}{\partial \mathbf{y}_{c}}\right) = \begin{cases} +, & \text{if } \mathbf{y}_{c} = 1, \\ -, & \text{otherwise} \end{cases} \tag{11}$$

where $\mathbf{y}_c$ is the multi-hot global label shared between M points. Thus, in contrast to the dense case in Eq. 10, CAM encourages a single point to be classified into multiple classes—despite a single point belonging to a single class—which results in false positive predictions.



CAM [50] **Ours** **Ours*** Ground Truth

Figure 6. **Additional qualitative results** – We show additional qualitative results of our and CAM [50] methods. We denote **Ours*** as training with bootstrapping.