GeoAuxNet: Towards Universal 3D Representation Learning for Multi-sensor Point Clouds

Shengjun Zhang[1], Xin Fei[2], Yueqi Duan[1†]
[1] Department of Electronic Engineering, Tsinghua University
[2] Department of Automation, Tsinghua University
{zhangsj23, feix21}@mails.tsinghua.edu.cn, duanyueqi@tsinghua.edu.cn

https://github.com/zhangshengjun2019/GeoAuxNet

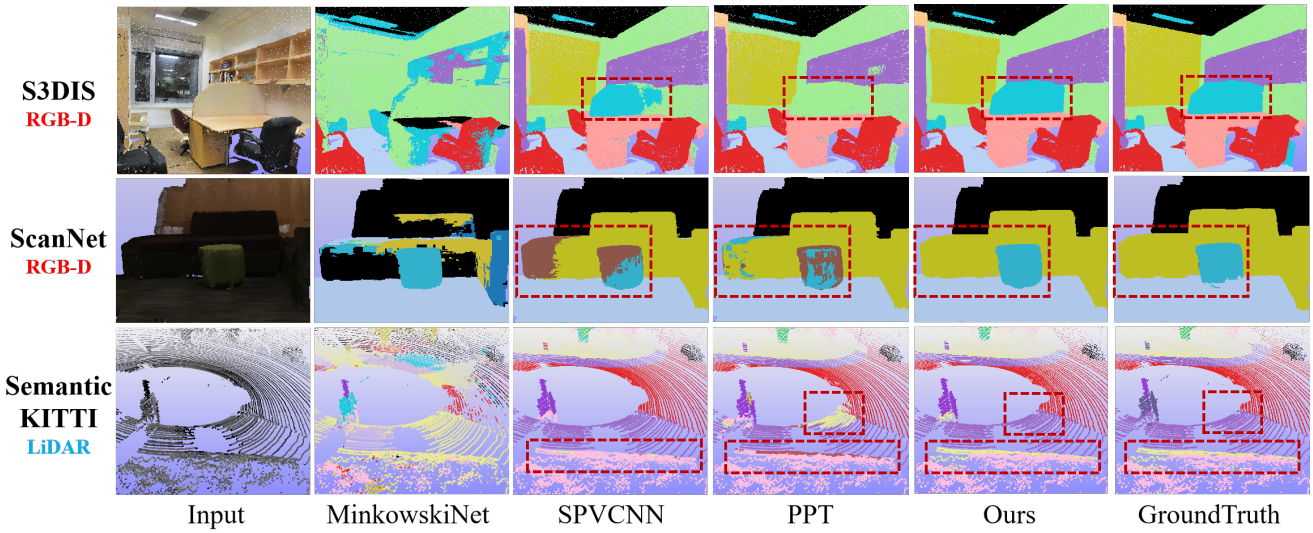


Figure 1. Semantic segmentation results on S3DIS [1] and ScanNet [9] from RGB-D cameras and SemanticKITTI [2] from LiDAR. For all methods, we trained collectively on three datasets. Our method outperforms other methods with better detailed structures.

Abstract

Point clouds captured by different sensors such as RGB-D cameras and LiDAR possess non-negligible domain gaps. Most existing methods design different network architectures and train separately on point clouds from various sensors. Typically, point-based methods achieve outstanding performances on even-distributed dense point clouds from RGB-D cameras, while voxel-based methods are more efficient for large-range sparse LiDAR point clouds. In this paper, we propose geometry-to-voxel auxiliary learning to enable voxel representations to access point-level geometric information, which supports better generalisation of the voxel-based backbone with additional interpretations of multi-sensor point clouds. Specifically, we construct hierarchical geometry pools generated by a voxel-guided dynamic point network, which efficiently provide auxiliary fine-grained geometric information adapted to different stages of voxel features. We conduct experiments on joint multi-sensor datasets to demonstrate the effectiveness of GeoAuxNet. Enjoying elaborate geometric information, our method outperforms other models collectively trained on multi-sensor datasets, and achieve competitive results with the-state-of-art experts on each single dataset.

1. Introduction

Point cloud analysis has attracted widespread attention due to its growing applications, such as autonomous driving [20, 45, 51] and robotics [4, 40]. Unlike images which are represented by regular pixels, 3D point clouds are irregular and unordered. These characteristics aggravate the inconsistency on density and sampling patterns of point clouds captured by different sensors, such as RGB-D cameras [1, 9, 33] and LiDAR [2, 5, 34].

†Corresponding author.

Typically, points generated from RGB-D pictures are equally distributed and dense, while points scanned by LiDAR are sparse and uneven. The diversity on input data hinders the construction of universal network architectures. For point clouds from RGB-D cameras, point-based methods [28, 29, 54] usually gather local information via grouping and aggregating neighborhood features to extract detailed spatial information. Since neighbors are not stored contiguously in point representations, indexing them requires the costly searching operations [24], which limits the application of these methods on large scale LiDAR point clouds. To address this challenge, voxel-based methods [12, 25, 31] leverage the regular 3D convolution to obtain the contiguous memory access pattern. However, the resolution of voxels is constrained by the memory and becomes lower at deeper stages, resulting in loss of local information [35]. An intuitive idea is to ensemble both point-based and voxel-based methods [24, 35, 52], yet they are still confronted with either high time-consumption or the lack of detailed geometric features, which fail to simultaneously process large scale point clouds from different sensors with fine-grained spatial information while maintaining efficiency.

In this paper, we propose GeoAuxNet to provide point-level geometric information for voxel representations in an efficient manner with geometry-to-voxel auxiliary learning. The support of elaborate geometric features introduces additional point-level information, which cannot be fully exploited by voxel-based backbones. Motivated by the observation that the local geometric structures present high similarity at each stage of the network, we construct hierarchical geometry pools to learn auxiliary sensor-aware point-level geometric priors corresponding to different layers for sensor-agnostic voxel features. To update our geometry pools, we present a voxel-guided dynamic point network, where we leverage prior knowledge in voxel features to guide high quality spatial feature extraction. Then, we fuse elaborate geometric features in the pools into voxel representations via our designed geometry-to-voxel auxiliary mechanism. For large-scale point clouds, the hierarchical geometry pools store representative point-level features at different stages, which efficiently provide complementary geometric information without using the point-based networks during inference time.

We conduct extensive experiments on multi-sensor datasets, including S3DIS [1] and ScanNet [9] from RGB-D cameras as well as SemanticKITTI [2] captured by LiDAR, to demonstrate the efficiency and effectiveness of our method. With a shared weight of backbone, our method outperforms other models trained on joint datasets from different sensors and achieves competitive performance with experts on single dataset. As shown in Figure 1, we trained MinkowskiNet [8], SPVCNN [35], PPT [43] and GeoAuxNet on the above three datasets collectively, where our method preserve better detailed structures for point clouds from various sensors.

2. Related Works

3D Scene Understanding. Since images are composed of regular pixels, point clouds generated from RGB-D cameras are well-distributed and dense. Voxelization on dense point clouds ignore detailed structure information. To directly learn from points, Qi *et al.* [27] first proposed PointNet to process point clouds through shared multi-layer perceptions (MLP) to get a global representation. Yet this method lacks the ability to comprehend local geometric features. Therefore, PointNet++ [28] built a hierarchical architecture to gather local information via grouping and aggregating neighborhood features. Following up MLP-based works [7, 29] focused on designing effective learning schemes to capture spatial features. Other methods also introduced specific convolution operations [3, 19, 36, 48], graph-based representations [22, 39] and self-attention mechanism [11, 16, 42, 49, 54] to explore the relationship between points. Apart from cameras, LiDAR is also a widely used sensor to collect point clouds. Some methods [15, 26, 41, 53] focused on projecting point clouds to 2D grids to utilize 2D CNN. However, 3D to 2D projection limits the representation quality of geometric structures [50] due to the sacrifice of 3D spatial information [48]. Point based methods have high time consumption because of the large scale and the sparsity of these point clouds. Voxel-based methods [12, 13, 17, 56, 57] show more robustness against the sparsity. In view of the sampling characteristic of LiDAR, Zhu *et al.* [56] introduced cylinder coordinate system, for nearby regions have greater density than farther away regions. Similarly, Lai *et al.* [17] proposed spherical transformer to aggregate information from dense close points to sparse distant ones. Voxel-based networks maintain efficiency when processing scene-level point clouds, but ignore some detailed spatial structures.

Universal Learning from Multiple Data Sources. Large language models achieve remarkable progress on many natural language processing systems through pretraining on extremely large text corpora [10], which enables foundation models to process data across domains, involving multiple languages, various majors, diverse application scenarios and so on. In 2D computer vision, joint learning across domains improve model robustness for detection [38, 47, 55], depth estimation [30] and semantic segmentation [18, 37]. Wang *et al.* [38] proposed a universal object detector through a domain-attention mechanism. But they did not model the training differences between different datasets. Universal-RCNN [47] modeled the class relations by designing a inter-dataset attention module to in-

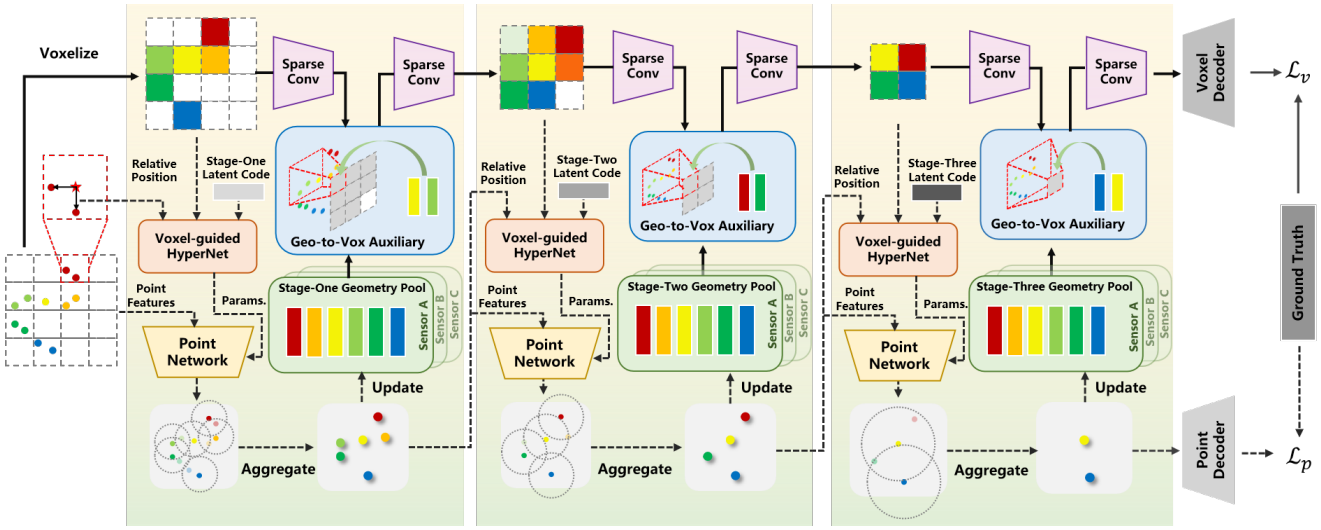


Figure 2. The pipeline of our GeoAuxNet. For a complete scene point cloud $\mathcal{P}^{\mathcal{C}}$ and a local point patch $\mathcal{P} \subseteq \mathcal{P}^{\mathcal{C}}$, our voxel-based backbone first voxelizes $\mathcal{P}^{\mathcal{C}}$ and conducts sparse convolutional operations. The voxel-guided hypernetwork takes relative positions, voxel features and a stage latent code as input to provide weights and biases for the point network. Then, we encode the spatial information for $\mathcal{P}$ with the point network and aggregate local features to generate geometric feature candidates. Following the update strategy, we construct hierarchical geometry pools. The geometry-to-voxel mechanism fuses geometric features stored in the pools to enable voxel representations to access point-level geometric information. We repeat the above process several times to extract effective representation hierarchically and predict the results with a voxel decoder for the primary task and a point decoder for the auxiliary task. The dotted line stands for the course of the auxiliary learning which is ignored during inference to ensure efficiency. Geo-to-Vox is abbreviation of Geometry-to-Voxel.

corporates graph transfer learning for propagating relevant semantic information across multiple datasets. CDCL [37] pointed out that convolution filters can be shared across domains without accuracy loss, while normalization operators are not appropriate to share in view of the bias of statistics. Recently, Wu *et al.* [43] introduced point prompt training to optimize a learnable domain-specific prompt with language-guided categorical alignment, but processed point clouds from RGB-D and LiDAR separately. However, point clouds captured by different sensors have diverse density and distributions, which limits the exploration of the universal network on multiple data sources.

Auxiliary Learning. Auxiliary learning aims to achieve outstanding performance on a single primary task with the assist of auxiliary tasks and has shown many successful applications in a wide range of fields, including knowledge distillation [46], transfer learning [14] and reinforcement learning [21]. For instance, DCAN [6] learned an auxiliary contour detection task for more accurate segmentation, while Araslanov investigated the joint learning of image classification and semantic segmentation. In 3D vision, MonoCon [23] introduced monocular contexts as auxiliary tasks in training for 3D object detection. Nevertheless, the inherent relationship between point-level features and voxel-level features is still under exploration to better leverage the advantages of two domains without any requirement of additional data.

3. Method

The overall framework of GeoAuxNet is illustrated in Figure 2. To generate point-level geometric features, we propose a voxel-guided hypernetwork to produce weights and biases for the point network. Then, point features are grouped and aggregated to extract fine-grained local features, which are employed to update geometry pools. For voxel-based backbone, we introduce geometry-to-voxel auxiliary mechanism to access elaborate spatial information in our hierarchical geometry pools.

3.1. Voxel-guided Dynamic Point Network

Point-based methods are mostly confronted with high time-consumption towards large scale point clouds, since the farthest point sampling (FPS) and k-nearest neighbors (kNN) algorithms in these networks have a time complexity of $\mathcal{O}(n^2)$. Therefore, to efficiently process large scale point clouds, input points are voxelized and passed through sparse convolution operations to extract voxel features in our voxel-based backbone.

To generate point-level geometric information efficiently, we train our point network on local point patches instead of complete scene point clouds. However, less training data weakens the ability of the point network to extract local features. Considering that our voxel-based branch are

trained on complete voxelized point clouds, we treat voxel features as prior knowledge to instruct the learning of local geometric information. Yet, points inside a voxel share the same voxel feature, which ignores detailed difference between points. Thus, we also introduce relative coordinates to depict diversity of points within the same voxel. On account of the hierarchical architectures of both the point network and the voxel-based backbone with several stages, we optimize a learnable latent code s for each stage during our auxiliary learning.

Given a local point patch $\mathcal{P} = \{p_i, 1 \leq i \leq N\} \in \mathbb{R}^{N\times 3}$ with point features $\mathcal{F}^{\mathcal{P}} = \{f_i^{\mathcal{P}}, 1 \leq i \leq N\}$, each point p_i belongs to a voxel v with the voxel feature $f^{\mathcal{V}}$. Here $v \in \mathbb{R}^3$ stands for the center coordinate of the voxel. Then, we can obtain the weight and bias for point p_i:

$$w_i = h_w(h_p(p_i - v), f^{\mathcal{V}}) \odot h_s(s), \tag{1}$$

$$b_i = h_b(h_p(p_i - v), f^{\mathcal{V}}) \odot h_s(s), \tag{2}$$

where h_p and h_s are MLPs to project relative positions and the stage latent code to specific dimensions, and h_w and h_b are hypernetworks to generate the weight and bias. Hence, new point features can be formulated as:

$$\hat{f}_i^{\mathcal{P}} = \text{MLP}(w_i \odot f_i^{\mathcal{P}} + b_i). \tag{3}$$

To extract local geometric information, we use kNN algorithm to select k nearest neighbor points and employ a max-pooling function to aggregate each local group. In this manner, we generate a set of geometric feature candidates for hierarchical geometry pools.

3.2. Hierarchical Geometry Pools

The voxel branch ignores some detailed spatial structures due to voxelization. Motivated by the similarity of geometric structures in 3D space, we establish a pool to store typical geometric patterns. The pool contains n typical spatial structure features $\mathcal{F}^{\mathcal{D}} = \{f_i^{\mathcal{D}}, 1 \leq i \leq n\}$. The max size of the pool is set to N. In Section 3.1, we generate a set of geometric pattern candidates $\mathcal{F}^{\mathcal{S}} = \{f_i^{\mathcal{S}}, 1 \leq i \leq m\}$. When updating the pool, we first calculate the similarity between $\mathcal{F}^{\mathcal{S}}$ and $\mathcal{F}^{\mathcal{D}}$:

$$s_i = \max_{1\leq j\leq N} s_{i,j}, \tag{4}$$

where

$$s_{i,j} = \frac{f_i^{\mathcal{S}} \cdot f_j^{\mathcal{D}}}{\|f_i^{\mathcal{S}}\| \|f_j^{\mathcal{D}}\|}. \tag{5}$$

Specifically, if $s_i \geq \epsilon$, $f_i^{\mathcal{S}}$ is merged into the geometry pool $\mathcal{F}^{\mathcal{D}}$:

$$t = \underset{1\leq j\leq N}{argmax}\, s_{i,j}, \tag{6}$$

$$\hat{f}_t^{\mathcal{D}} = \lambda f_t^{\mathcal{D}} + (1-\lambda) f_i^{\mathcal{S}}, \tag{7}$$

Algorithm 1 Update Pools

Input: New features $\mathcal{F}^{\mathcal{S}}$, geometry pool $\mathcal{F}^{\mathcal{D}}$, maximum size of the pool N, threshold ϵ, update rate λ
Output: New pool $\hat{\mathcal{F}}^{\mathcal{D}}$

```
Compute s_i for 1 ≤ i ≤ m according to Eq. 4
for i ← 1 to |F^S| do
    if s_i ≥ ε then
        t ← arg max_{1≤j≤N} s_{i,j}
        f_t^D ← λ f_t^D + (1 − λ) f_i^S
        F^S ← F^S \ {f_i^S}
    end if
end for
if |F^S| + |F^D| ≤ N then
    F^D ← F^D ∪ F^S
else
    F̂^S ← randomly select N − |F^D| features from F^S
    F^S ← F^S \ F̂^S
    F^D ← F^D ∪ F̂^S
    for i ← 1 to |F^S| do
        t ← arg max_{1≤j≤N} s_{i,j}
        f_t^D ← λ f_t^D + (1 − λ) f_i^S
    end for
end if
```

where ϵ is a threshold and λ stands for the update rate. Otherwise, we append the new features to the geometry pool. If the total number of features is more than N, we randomly select $N - n$ new features and add them to the pool. The rest new features following Eq. 7 are merged into existing patterns. Our update strategy is illustrated in Algorithm 1.

Since voxel representations at various stages contains different level information, we introduce hierarchical geometry pools to store representative point-level features corresponding to different stages. We also enlarge the size of pools for deeper stages due to more complex geometric structures with larger receptive field sizes.

3.3. Geometry-to-Voxel Auxiliary

After we update the geometric pool, we use geometric patterns stored in $\mathcal{F}^{\mathcal{D}} \in \mathbb{R}^{N\times C}$ to generate fine-grained local information for each voxel following a cross-attention operation. Given voxel features $\mathcal{F}^{\mathcal{V}} = \{f_i^{\mathcal{V}}, 1 \leq i \leq M\} \in \mathbb{R}^{M\times D}$, the geometric fine-grained feature $f_i^{\mathcal{G}}$ for $f_i^{\mathcal{V}} \in \mathcal{F}^{\mathcal{V}}$ is formulated by:

$$q_i = h_Q(f_i^{\mathcal{V}}) \in \mathbb{R}^{1\times s}, \tag{8}$$

$$K = h_K(\mathcal{F}^{\mathcal{D}}) \in \mathbb{R}^{N\times s}, \tag{9}$$

$$V = h_V(\mathcal{F}^{\mathcal{D}}) \in \mathbb{R}^{N\times t}, \tag{10}$$

$$f_i^{\mathcal{G}} = softmax\left(\frac{q_i K^\top}{\sqrt{s}}\right) V \in \mathbb{R}^{1\times t}, \tag{11}$$

where h_Q, h_K, and h_V stand for projections. Then, we concatenate $f_i^{\mathcal{G}}$ with $f_i^{\mathcal{V}}$ as the new voxel feature and employ sparse convolution operations. In this manner, we can provide complementary geometric information without using the point-based networks during inference stage.

3.4. Overall Architecture

Given a point cloud $\mathcal{P}_i^{\mathcal{C}}$ with features $\mathcal{F}_i^{\mathcal{C}}$ and a local point patch $\mathcal{P}_i \subseteq \mathcal{P}_i^{\mathcal{C}}$ with $\mathcal{F}_i^{\mathcal{P}} \subseteq \mathcal{F}_i^{C}$ from sensor $\mathcal{S}_i$, we voxelize all points to obtain voxel coordinates $\mathcal{V}_i$ with voxel features $\mathcal{F}_i^{\mathcal{V}}$. Points obtained from RGB-D cameras usually possess colors and normal vectors, while points captured by LiDAR always possess intensity. Thus, we introduce sensor-aware input embedding. Specifically, we embed $\mathcal{P}_i$ from sensor $\mathcal{S}_i$ by:

$$\hat{\mathcal{F}}_i^{\mathcal{P}} = h_{\mathcal{S}_i}^{\mathcal{P}}(\mathcal{F}_i^{P}), \tag{12}$$

$$\hat{\mathcal{F}}_i^{\mathcal{V}} = h_{\mathcal{S}_i}^{\mathcal{V}}(\mathcal{F}_i^{V}), \tag{13}$$

where $h_{\mathcal{S}_i}^{\mathcal{P}}$ means shared MLPs on points and $h_{\mathcal{S}_i}^{\mathcal{V}}$ means sparse convolution operations on voxels. Then, we learn voxel features and point features following the extraction strategy in Section 3.1 and update hierarchical geometry pools using Algorithm 1. We fuse point-level geometric information to voxel representation via geometry-to-voxel auxiliary mechanism. Our auxiliary task is to predict point labels for local point patches with point branch and point decoder. Therefore, the final loss function is formulated as:

$$\mathcal{L} = \mathcal{L}_v + \mu \mathcal{L}_p, \tag{14}$$

where $\mathcal{L}_v$ and $\mathcal{L}_p$ stand for per-voxel and per-point cross entropy loss and μ is the weight of the point loss.

3.5. Discussion

In this section, we highlight previous point-voxel networks [24, 35, 52] which combine a point-based branch and a voxel-based branch together to leverage the advantages of both efficient voxel representations and elaborate point features. Additionally, we compare our GeoAuxNet with these methods. PVCNN [24] and PVT [52] rely on point-based networks to extract high quality representations with time-consuming searching or attention mechanisms, while SPVCNN [35] does not explore detailed local geometric information. Since the interaction between two branches in these methods only contains voxelization and devoxelization operations, they fail to benefit from both voxel representations and point-level elaborate features. Our GeoAuxNet preserves hierarchical sensor-aware geometry pools as the bridge to enable voxel representations to absorb point-level local geometric information without using the point network during inference.

4. Experiments

We validate our proposed GeoAuxNet across point clouds from multiple sensors. In Section 4.1, we train various methods on joint datasets including S3DIS [1], ScanNet [9] and SemanticKITTI [2]. In Section 4.2, we analyze the effectiveness of our sensor-aware hierarchical geometry pools. In Section 4.3, we compare GeoAuxNet with typical point-based and voxel-based methods to demonstrate the efficiency of our designed geometry-to-voxel auxiliary learning. In Section 4.4, we ablate different design choices of our voxel-guided dynamic point network.

4.1. Semantic Segmentation

Dataset. We conduct semantic segmentation experiments on the joint data of three datasets: S3DIS [1] and ScanNet [9], which are generated from RGB-D cameras, and SemanticKITTI [2], which is captured by LiDAR. The S3DIS dataset comprises 271 rooms from six areas in three distinct buildings, which are densely sampled on the surfaces of the meshes and annotated into 13 categories. Model performance evaluation is typically done using results from Area 5. The ScanNet dataset comprises 1,613 scene scans reconstructed from RGB-D frames. It is divided into 1,201 scenes for training, 312 scenes for validation, and 100 scenes for benchmark testing. SemanticKITTI consists of 22 point cloud sequences, where sequences 00 to 10, 08 and 11 to 21 are used for training, validation and testing. After merging classes with distinct moving status and discarding classes with very few points, a total of 19 classes are selected for training and evaluation.

Implementation Details. Our voxel-based backbone is built on MinkowskiNet [8] with the batch normalization operations proposed by PPT [43]. The voxel size is set to 0.05m. We train our methods as well as MinkowskiNet SPVCNN [35] and PPT [43] on joint input data [1, 2, 9] captured by RGB-D cameras and LiDAR collectively. We also fine tune our method on three datasets separately. For the sake of fairness, we set the number of parameters of MinkowskiNet, SPVCNN, PPT and GeoAuxNet to about 60M. We use SGD optimizer and OneCycleLR [32] scheduler, with learning rate 0.05 and weight decay 10^{-4}. We adopt OneCycleLR [32] scheduler with 5% percentage of the cycle spent increasing the learning rate and a cosine annealing strategy. We train all models with data augmentation used in PPT [43]. We set λ in Eq. 7 to 0.1, μ in Eq. 14 to 0.1, threshold for geometry pool update ϵ to 0.9. The total number of iterations is equal to the sum of necessary iterations for all datasets. The sampling ratio across S3DIS, ScanNet and SemanticKITTI is set to 2:2:5. The number of points for the auxiliary branch does not exceed 2×10^4. The max size of hierarchical geometry pools for each stage

Table 1. Semantic segmentation results on three benchmarks including S3DIS [1], ScanNet [9] and SemanticKITTI [2]. We train SPVCNN [35], PPT [43] and GeoAuxNet on the joint training data of three datasets and also compare with experts [16, 17, 29, 35, 54, 56] on each single dataset. We report the class-average accuracy (mAcc, %) and class-wise mean IoU (mIoU, %). The type stands for point-based (p) or voxel-based (v) methods, where $p \rightarrow v$ means the geometry-to-voxel auxiliary learning.

Method	Type	Params.	S3DIS Area5		ScanNet		SemanticKITTI	
			Test mIoU	Test mAcc	Val mIoU	Val mAcc	Val mIoU	Val mAcc
PointTransformer [54]	p	11.4M	70.4	76.5	70.6	-		
StratifiedFromer [16]	p	18.8M	72.0	78.1	74.3	-		
PointNeXt [29]	p	41.6M	70.5	77.2	71.5	-		
SPVCNN [35]	$p+v$	21.8M					63.8	-
Cylinder3D [56]	v	26.1M					64.3	-
SphereFormer [17]	v	32.3M					67.8	-
MinkowskiNet [8]	v	60.9M	18.8	21.3	25.6	30.5	36.0	41.2
SPVCNN [35]	$p+v$	61.0M	58.6	62.3	56.7	60.4	52.0	56.7
PPT [43]	v	63.0M	30.4	34.6	16.5	27.4	25.0	31.2
PPT † [43]	v	63.0M	67.9	73.2	69.6	77.5	61.8	68.9
GeoAuxNet	$p \rightarrow v$	64.7M	**69.5**$_{(+1.6)}$	**74.5**$_{(+1.3)}$	**71.3**$_{(+1.7)}$	**79.3**$_{(+1.8)}$	**63.8**$_{(+2.0)}$	**69.3**$_{(+0.4)}$

† The original PPT with language-guided categorical alignment fails to converge on the joint training data. We use decoupled projection heads to employ PPT on multi-sensor datasets.

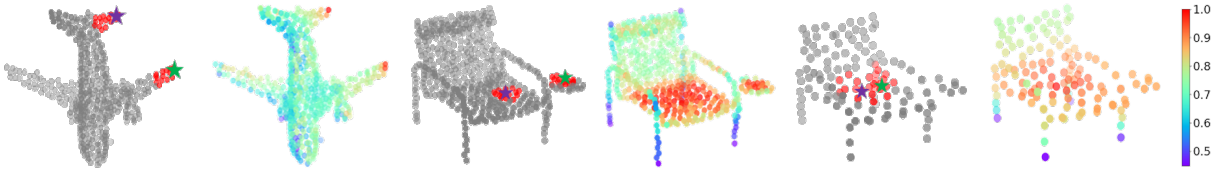
is set to 32, 64, 128, 256 separately.

Results Comparison. As shown in Table 1, GeoAuxNet surpasses other methods which are trained on three datasets jointly. Our methods outperforms SPVCNN [35], which does not utilize its point branch to extract local information, by more than 10% in mIoU on each dataset. PPT [43] introduces different datasets as prompts to collaboratively train a single model on multiple datasets from the same sensor type. The language-guided alignment of PPT suffers from the large domain gap caused by different sensors, which fails to converge on the joint training datasets. Therefore, we replace it by the decoupled alignment with separate heads for each dataset, which is also defined in PPT [43]. We construct sensor-aware geometry pools to address this issue and outperform PPT by about 2% in mIoU on each dataset. Besides, we also compare with experts trained on each single dataset. We achieve competitive results with point-based methods on datasets from RGB-D cameras. More specifically, we even outperform PointTransformer in validation mIoU of ScanNet. Despite training on inter-domain datasets from different sensors, GeoAuxNet has a higher accuracy on SemanticKITTI than SPVCNN [35], and attain encouraging performances even compared with recent state-of-the-art expert [17]. Although we do not introduce any specific designs such as point-wise attention mechanism [16, 54] and cylinder coordinate system [56] for sparse LiDAR point clouds, the quantitative results demonstrate the effectiveness of geometry-to-voxel auxiliary learning by constructing hierarchical geometry pools to provide auxiliary sensor-aware point-level geometric priors for voxel representations.

4.2. Hierarchical Geometry Pools

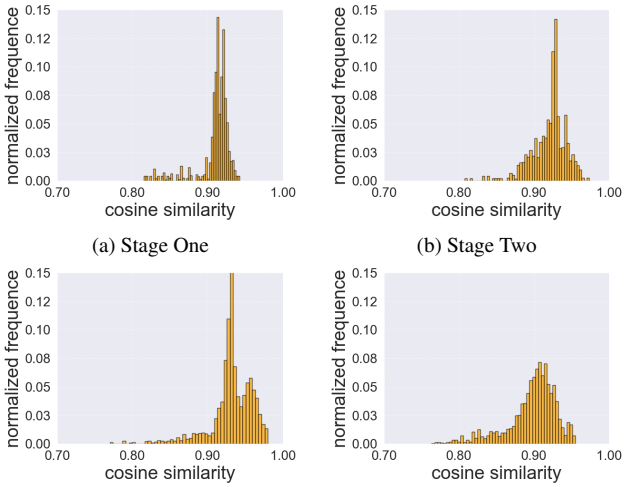
In this section, we further conduct experiments on our hierarchical geometry pools. First, we analyze the similarity of geometric structures in 3D space. Second, we discuss the appropriation to represent geometric information with our geometry pools. Finally, we study the construction of sensor-aware pools for different sensors.

Similarity of Geometric Structures. Local geometric structures possess similarity in 3D space. For example, planar structures exist in desks, chairs and airplanes. These spatial structures are not unrelated but consistent. To further analyze the similarity of geometric structures learned by networks, we train our point encoder on ModelNet40 [44]. For each point, we use kNN to search 16 nearest neighbors and aggregate them via a max-pooling function to generate local geometric features. We select a center point marked as a purple star in Figure 3a and Figure 3c, and calculate the cosine similarity between its features and other point features. Specifically, the wing and the tail of the plane in Figure 3a have similar local structures, resulting in high similarity of features in Figure 3b. Figure 3d and Figure 3f show the same phenomenon at different stages in the network, where planar structures are similar to each other, while different from the chair legs. Since features from deeper lay-



(a) (b) (c) (d) (e) (f)

Figure 3. Visualization of the cosine similarity between features. The purple star is the selected point, and the green star is the point with a significantly similar feature to the purple star. We calculate the cosine similarity between the feature of the red star and other features and visualize them in image (b), (d) and (f). The nearest neighbors of the purple and green stars are marked red in image (a), (c) and (e). (a), (b), (c) and (e) are generated from the first stage of the point network and (e), (f) are from the second stage.



(a) Stage One (b) Stage Two

(c) Stage Three (d) Stage Four

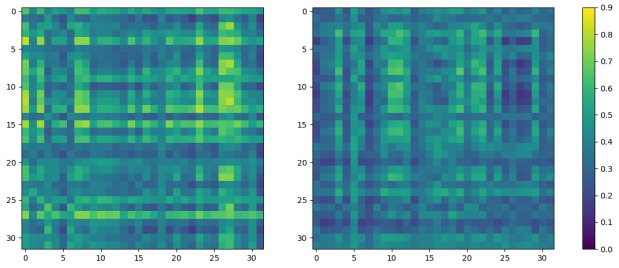
Figure 4. Statistical results of cosine similarity between hierarchical geometry pools and point-level features extracted by the point network from point clouds in S3DIS [1] at different stages.

ers possess larger receptive field sizes due to grouping and aggregating operations and comprise more geometric information, our hierarchical pools contain more geometric features for deeper layers.

Representativeness of Geometry Pools. To demonstrate the representativeness of our pools, we calculate cosine similarity between all features in a scene from S3DIS [1] extracted by the point encoder and the features stored in the pool. For each stage, given the geometry pool $\mathcal{F}^D = \{f_i^D, 1 \leq i \leq N\}$ and all features in a scene $\mathcal{F}^S = \{f_i^S, 1 \leq i \leq M\}$, the similarity s_i between f_i^S and $\mathcal{F}^D$ is formulated as:

$$s_i = \max_{1 \leq j \leq N} \frac{f_i^S \cdot f_j^D}{\|f_i^S\| \|f_j^D\|}, \tag{15}$$

where $\|\cdot\|$ stands for L2 norm. The distribution of similarities is shown in Figure 4. From the statistical data, most features have cosine similarity higher than 0.85, which indicates that each feature can find a similar feature among our



(a) (b)

Figure 5. Cosine similarity between geometry pools. (a) shows the similarity between geometry pools of S3DIS [1] and ScanNet [9]. (b) shows the similarity between geometry pools of S3DIS [1] and SemanticKITTI [2]. Intra-sensor geometry pools in (a) have higher similarity than inter-sensor geometry pools in (b).

learned representatives in the pool. In this manner, we avoid using point-based network to process large scale points to generate fine-grained features for each voxel. Our hierarchical pools contain typical geometric features of scenes. Therefore, we can fuse point-level geometry information stored in our pools into voxel representations via geometry-to-voxel auxiliary mechanism efficiently.

Sensor-aware Pools for Multiple Sensors. Motivated by the diversity of density and sampling patterns of point clouds from various sensors, we construct different geometry pools to preserve relevant geometric information for diverse sensors. We investigate another two strategies: (i) shared geometry pools across sensors and (ii) different geometry pools for different datasets. We calculate the cosine similarity between features from different pools. As shown in Figure 5, geometry pools from the same sensor possess higher similarity than those from different sensors. As shown in Table 2, training with shared geometry pools across sensors leads to a significant decrease on performance due to the inherent diversity of geometric structures from different sensors, while sharing geometry pools with datasets from the same sensor serves as data augmentation to improve the effectiveness of the geometry pools.

Table 2. Ablation study of sensor-aware geometry pools. "Shared intra-sensor" means the geometry pools which are shared for point clouds from the same sensor, while different for point clouds from different sensors. "Shared inter-sensor" means the geometry pools shared for different sensors. "Shared intra-dataset" means the geometry pools are different for different datasets. We report mIoU and mAcc on Area 5 of S3DIS [1].

Methods	mIoU (%)	mAcc (%)
Shared intra-sensor	69.5	74.5
Shared inter-sensor	$61.4_{(-8.1)}$	$68.3_{(-6.2)}$
Shared intra-dataset	$67.5_{(-2.0)}$	$73.3_{(-1.2)}$

Table 3. Efficiency analysis. We report the inference time and throughput on pre-processed data from S3DIS dataset.

Method	Params.	Inference Time (ms)	Throughput (ins./sec.)
PointNet++ [28]	1.0M	378.7	108
PT [54]	7.8M	1079.7	34
PTv2 [42]	3.9M	4732.2	12
MinkowskiNet [8]	60.9M	45.4	898
SPVCNN [35]	61.0M	48.4	682
PPT [43]	63.0M	80.4	251
GeoAuxNet	64.7M	51.8	572

4.3. Efficiency of GeoAuxNet

We validate the efficiency of GeoAuxNet compared with other typical point-based [42, 54] and voxel-based methods [8, 43]. We focus on the mean inference time and throughput by employing a standardized pre-processed point cloud with instance labels from S3DIS Area 5 dataset as the input on a single NVIDIA A40 GPU. Initially, the device undergoes a "warm-up" phase, where the model is fed with the input point cloud ten times consecutively to eliminate any latency associated with data loading. Subsequently, the model is subjected to an additional 300 iterations of input feeding to calculate mean inference time. To effectively quantify the throughput, we measure the number of instances that a model processes per second. This systematic approach ensures a thorough and reliable assessment of our GeoAuxNet and other state-of-the-art methods.

In Table 3, we show the number of parameters, inference time and throughput for typical point-based and voxel-based networks. Specifically, our GeoAuxNet outperforms the expert model of PointTransformer [54] which is trained on ScanNet [9] individually in mIoU with $16.8\times$ throughput and $20.8\times$ inference speedup. Even compared with the simple PointNet++ [28] whose parameter size is $1/68$ of ours, GeoAuxNet has less inference time and higher throughput. Besides, our method has similar inference time and throughput to voxel-based methods but achieve better performance according to the results in Table 1. GeoAuxNet surpasses PPT [43] on each benchmark about 2% in mIoU with $2.3\times$ throughput and $1.6\times$ inference speedup. Our method maintains efficiency for large scale point clouds, while achieves encouraging performances on datasets from multiple sensors.

Table 4. Ablation study of our voxel-guided hypernetwork. We report the mIoU and mAcc on Area 5 of S3DIS [1].

Methods	mIoU (%)	mAcc (%)
Vanilla	63.3	68.7
+ voxel-guided	$66.0_{(+2.7)}$	$72.5_{(+3.8)}$
+ relative positions	$66.8_{(+0.8)}$	$73.2_{(+0.7)}$
+ stage latent code	$67.6_{(+0.8)}$	$73.3_{(+0.1)}$

4.4. Voxel-guided Hypernetworks

To investigate the architecture design of our voxel-guided hypernetwork, we conduct ablation studies on S3DIS semantic segmentation. We first introduce a vanilla point network which learns the weight and bias directly. Then, we adopt a hypernetwork taking voxel features as input to generate weights and biases for the point network. We further take account of the relative positions of points. Finally, we optimize learnable stage latent codes to instruct the information of different layers. We train these four models on S3DIS dataset. As shown in Table 4, the exploration of voxel features provides significant guidance for the point network to learn high quality geometric information, leading to an increase in test accuracy by 2.7% in mIoU and 3.8% in mAcc. Relative positions and stage latent codes also improve the performance of our methods.

5. Conclusion

In this paper, we propose GeoAuxNet for multi-sensor point clouds with designed geometry-to-voxel auxiliary learning. We construct hierarchical geometry pools to learn auxiliary sensor-aware point-level geometric priors at different layers for sensor-agnostic voxel features. To generate our geometry pools, we also introduce a voxel-guided dynamic point network to leverage voxel prior knowledge for elaborate point features extraction. Our proposed geometry-to-voxel auxiliary builds a bridge between point-level and voxel-level features in an efficient manner without using the point network during inference time. Experimental results have shown the effectiveness and efficiency of our methods. We hope this work will inspire future research towards universal representation learning for point clouds.

Acknowledgements. This work was supported by the National Natural Science Foundation of China under Grant 62206147.

References

[1] Iro Armeni, Ozan Sener, Amir Roshan Zamir, Helen Jiang, Ioannis K. Brilakis, Martin Fischer, and Silvio Savarese. 3D Semantic Parsing of Large-Scale Indoor Spaces. *CVPR*, pages 1534–1543, 2016. 1, 2, 5, 6, 7, 8

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, C. Stachniss, and Juergen Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. *ICCV*, pages 9296–9306, 2019. 1, 2, 5, 6, 7

[3] Alexandre Boulch. ConvPoint: Continuous convolutions for point cloud processing. *CGF*, 88:24–34, 2019. 2

[4] César Cadena, Anthony R. Dick, and Ian D. Reid. Multimodal Auto-Encoders as Joint Estimators for Robotics Scene Understanding. In *RSS*, 2016. 1

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yuxin Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. *CVPR*, pages 11618–11628, 2019. 1

[6] Hao Chen, Xiaojuan Qi, Lequan Yu, and Pheng-Ann Heng. DCAN: Deep Contour-Aware Networks for Accurate Gland Segmentation. *CVPR*, pages 2487–2496, 2016. 3

[7] Jaesung Choe, Chunghyun Park, Francois Rameau, Jaesik Park, and In So Kweon. PointMixer: MLP-Mixer for Point Cloud Understanding. *ECCV*, 2022. 2

[8] Christopher Bongsoo Choy, JunYoung Gwak, and Silvio Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. *CVPR*, pages 3070–3079, 2019. 2, 5, 6, 8

[9] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. *CVPR*, pages 2432–2443, 2017. 1, 2, 5, 6, 7, 8

[10] Jesse Dodge, Ana Marasovic, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting Large Webtext Corpora: A Case Study on the Colossal Clean Crawled Corpus. In *EMNLP*, 2021. 2

[11] Nico Engel, Vasileios Belagiannis, and Klaus C. J. Dietmayer. Point Transformer. *IEEE*, 9:134826–134840, 2020. 2

[12] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. *CVPR*, pages 9224–9232, 2018. 2

[13] Lei Han, Tian Zheng, Lan Xu, and Lu Fang. OccuSeg: Occupancy-Aware 3D Instance Segmentation. *CVPR*, pages 2937–2946, 2020. 2

[14] Judy Hoffman, Eric Tzeng, Trevor Darrell, and Kate Saenko. Simultaneous Deep Transfer Across Domains and Tasks. *ICCV*, pages 4068–4076, 2015. 3

[15] Lingdong Kong, You-Chen Liu, Runnan Chen, Yuexin Ma, Xinge Zhu, Yikang Li, Yuenan Hou, Y. Qiao, and Ziwei Liu. Rethinking Range View Representation for LiDAR Segmentation. *ArXiv*, abs/2303.05367, 2023. 2

[16] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified Transformer for 3D Point Cloud Segmentation. *CVPR*, pages 8490–8499, 2022. 2, 6

[17] Xin Lai, Yukang Chen, Fanbin Lu, Jianhui Liu, and Jiaya Jia. Spherical Transformer for LiDAR-Based 3D Recognition. *CVPR*, pages 17545–17555, 2023. 2, 6

[18] John Lambert, Zhuang Liu, Ozan Sener, James Hays, and Vladlen Koltun. MSeg: A Composite Dataset for Multi-Domain Semantic Segmentation. *PAMI*, 45:796–810, 2021. 2

[19] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution On X-Transformed Points. In *NIPS*, 2018. 2

[20] Yingwei Li, Adams Wei Yu, Tianjian Meng, Benjamin Caine, Jiquan Ngiam, Daiyi Peng, Junyang Shen, Bo-Xun Wu, Yifeng Lu, Denny Zhou, Quoc V. Le, Alan Loddon Yuille, and Mingxing Tan. DeepFusion: Lidar-Camera Deep Fusion for Multi-Modal 3D Object Detection. *CVPR*, pages 17161–17170, 2022. 1

[21] Xingyu Lin, Harjatin Singh Baweja, George A. Kantor, and David Held. Adaptive Auxiliary Task Weighting for Reinforcement Learning. In *NIPS*, 2019. 3

[22] Zhi-Hao Lin, Sheng Yu Huang, and Y. Wang. Convolution in the Cloud: Learning Deformable Kernels in 3D Graph Convolution Networks for Point Cloud Analysis. *CVPR*, pages 1797–1806, 2020. 2

[23] Xianpeng Liu, Nan Xue, and Tianfu Wu. Learning Auxiliary Monocular Contexts Helps Monocular 3D Object Detection. In *AAAI*, 2021. 3

[24] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-Voxel CNN for Efficient 3D Deep Learning. In *NIPS*, 2019. 2, 5

[25] Daniel Maturana and Sebastian A. Scherer. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. *IROS*, pages 922–928, 2015. 2

[26] Andres Milioto, Ignacio Vizzo, Jens Behley, and C. Stachniss. RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. *IROS*, pages 4213–4220, 2019. 2

[27] C. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CVPR*, pages 77–85, 2017. 2

[28] C. Qi, L. Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NIPS*, 2017. 2, 8

[29] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies. In *NIPS*, 2022. 2, 6

[30] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer. *PAMI*, 44:1623–1637, 2019. 2

[31] Gernot Riegler, Ali O. Ulusoy, and Andreas Geiger. OctNet: Learning Deep 3D Representations at High Resolutions. *CVPR*, pages 6620–6629, 2017. 2

[32] Leslie N. Smith and Nicholay Topin. Super-Convergence: Very Fast Training of Residual Networks Using Large Learning Rates. *ArXiv*, abs/1708.07120, 2017. 5

[33] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. *CVPR*, pages 567–576, 2015. 1

[34] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott M. Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. *CVPR*, pages 2443–2451, 2019. 1

[35] Haotian* Tang, Zhijian* Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In *ECCV*, 2020. 2, 5, 6, 8

[36] Hugues Thomas, C. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. *ICCV*, pages 6410–6419, 2019. 2

[37] Li Wang, Dong Wei Li, Han Liu, Jinzhan Peng, Lu Tian, and Yi Shan. Cross-Dataset Collaborative Learning for Semantic Segmentation. *AAAI*, 2021. 2, 3

[38] Xudong Wang, Zhaowei Cai, Dashan Gao, and Nuno Vasconcelos. Towards Universal Object Detection by Domain Attention. *CVPR*, pages 7281–7290, 2019. 2

[39] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic Graph CNN for Learning on Point Clouds. *TOG*, 38:1 – 12, 2018. 2

[40] Christian Wojek, Stefan Walk, Stefan Roth, and Bernt Schiele. Monocular 3D scene understanding with explicit occlusion reasoning. *CVPR*, pages 1993–2000, 2011. 1

[41] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud. *ICRA*, pages 1887–1893, 2018. 2

[42] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point Transformer V2: Grouped Vector Attention and Partition-based Pooling. In *NIPS*, 2022. 2, 8

[43] Xiaoyang Wu, Zhuotao Tian, Xin Wen, Bohao Peng, Xihui Liu, Kaicheng Yu, and Hengshuang Zhao. Towards Large-scale 3D Representation Learning with Multi-dataset Point Prompt Training. *ArXiv*, abs/2308.09718, 2023. 2, 3, 5, 6, 8

[44] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A deep representation for volumetric shapes. *CVPR*, pages 1912–1920, 2015. 6

[45] Yichen Xie, Chenfeng Xu, Marie-Julie Rakotosaona, Patrick Rim, Federico Tombari, Kurt Keutzer, Masayoshi Tomizuka, and Wei Zhan. SparseFusion: Fusing Multi-Modal Sparse Representations for Multi-Sensor 3D Object Detection. *ICCV*, abs/2304.14340, 2023. 1

[46] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge Distillation Meets Self-Supervision. In *ECCV*, 2020. 3

[47] Hang Xu, Linpu Fang, Xiaodan Liang, Wenxiong Kang, and Zhenguo Li. Universal-RCNN: Universal Object Detector via Transferable Graph R-CNN. In *AAAI*, 2020. 2

[48] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. PAConv: Position Adaptive Convolution with Dynamic Kernel Assembling on Point Clouds. *CVPR*, pages 3172–3181, 2021. 2

[49] Yu-Qi Yang, Yu-Xiao Guo, Jiangfeng Xiong, Yang Liu, Hao Pan, Peng-Shuai Wang, Xin Tong, and Baining Guo. Swin3D: A Pretrained Transformer Backbone for 3D Indoor Scene Understanding. *ArXiv*, abs/2304.06906, 2023. 2

[50] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: Sparse-to-Dense 3D Object Detector for Point Cloud. *ICCV*, pages 1951–1960, 2019. 2

[51] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3D Object Detection and Tracking. *CVPR*, pages 11779–11788, 2021. 1

[52] Cheng Zhang, Haocheng Wan, Xinyi Shen, and Zizhao Wu. PVT: Point-voxel transformer for point cloud learning. *IJIS*, 37:11985 – 12008, 2021. 2, 5

[53] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, and Hassan Foroosh. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. *CVPR*, pages 9598–9607, 2020. 2

[54] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point Transformer. In *ICCV*, 2021. 2, 6, 8

[55] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Simple Multi-dataset Detection. *CVPR*, pages 7561–7570, 2021. 2

[56] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and Asymmetrical 3D Convolution Networks for LiDAR Segmentation. *CVPR*, pages 9934–9943, 2021. 2, 6

[57] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In *MICCAI*, 2016. 2