# LTA-PCS: Learnable Task-Agnostic Point Cloud Sampling

Jiaheng Liu[1*], Jianhao Li[1*], Kaisiyuan Wang[3], Hongcheng Guo[1], Jian Yang[1],
Junran Peng[2], Ke Xu[1], Xianglong Liu[1], Jinyang Guo[1†]
[1]Beihang University [2]Institute of Automation, Chinese Academy of Sciences
[3]The University of Sydney

## Abstract

*Recently, many approaches directly operate on point clouds for different tasks. These approaches become more computation and storage demanding when point cloud size is large. To reduce the required computation and storage, one possible solution is to sample the point cloud. In this paper, we propose the first Learnable Task-Agnostic Point Cloud Sampling (LTA-PCS) framework. Existing task-agnostic point cloud sampling strategy (e.g., FPS) does not consider semantic information of point clouds, causing degraded performance on downstream tasks. While learning-based point cloud sampling methods consider semantic information, they are task-specific and require task-oriented ground-truth annotations. So they cannot generalize well on different downstream tasks. Our LTA-PCS achieves task-agnostic point cloud sampling without requiring task-oriented labels, in which both the geometric and semantic information of points is considered in sampling. Extensive experiments on multiple downstream tasks demonstrate the effectiveness of our LTA-PCS.*

## 1. Introduction

3D point cloud has been widely used in many areas like autonomous driving and robotics. However, the point clouds collected by LiDAR or other sensors contain millions or even billions of points, making it challenging to process, store, and transmit them efficiently. Therefore, to improve computational efficiency and reduce memory costs, it is critical to develop point cloud sampling methods to reduce the number of points while preserving the essential features of the original point clouds.

Among point cloud sampling methods, the most widely used one is the farthest point sampling (FPS) strategy in Fig. 1(a). It samples the critical points based on their 3D coordinates iteratively and selects a group of points that are farthest apart from each other. Although FPS is task-agnostic, it does not utilize the semantic information of point clouds as the sampling process is solely based on the point coordinates, causing degraded performance on downstream tasks. Recently, in Fig. 1 (b), learning-based point cloud sampling methods [7, 15] are proposed to sample the

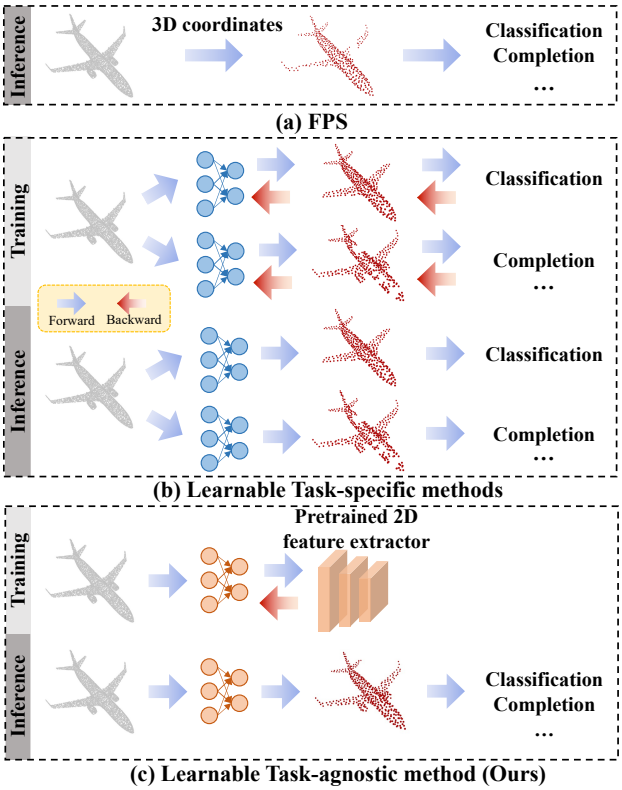


Figure 1. Comparison of FPS, learnable task-specific methods, and our LTA-PCS. FPS samples points using 3D coordinates, which ignores semantic meaning. Learnable Task-specific methods need different sampling networks for different tasks. Our LTA-PCS is task-agnostic, where one trained sampling network can be used for different tasks. Note that the pretrained 2D feature extractor is frozen in training, and not used at inference.

* First two authors contributed equally.
† Corresponding author.

points with more semantic meanings for better downstream task performance. They additionally introduce task-specific losses when learning the network for point cloud sampling. However, these methods are task-specific and require task-oriented labels. In real-world applications, the downstream tasks are usually not specified. But we need to sample fewer points in the collected point cloud by LiDAR or other sensors to save the storage. Moreover, even downstream tasks are given, the existing learning-based methods need to sample the points again based on the new downstream task for better performance, which is cumbersome in practice. Besides, the existing learning-based methods use the pretrained 3D feature extractor (e.g., PointNet [27]) to calculate the task loss for training the sampling network, while the capacity of these 3D feature extractors is limited by both model size and the scale of 3D point cloud datasets, which becomes the bottleneck for training the sampling network with semantically meaningful sampling ability. Finally, the task-specific loss makes them prone to overfit on specific datasets and hard to generalize on different datasets, and the requirement of labels increases the training cost. Although we can remove the task-specific loss in these learning-based methods for task-agnostic point cloud sampling, the sampled points still lack semantic meaning in this case.

So the problem is: *How to design a learnable task-agnostic label-free point cloud sampling strategy with the consideration of semantic information?* To solve this problem, in Fig. 1(c), we propose the first Learnable Task-Agnostic Point Cloud Sampling (LTA-PCS) framework, in which the semantic meaning of points is considered in the sampling process. It is non-trivial to construct such a framework. Without the specific downstream tasks, the designed sampling loss for learning-based sampling methods is based on the coordinates, which lack semantic meaning.

In LTA-PCS, we observe 2D feature extractors [31] are effective tools to extract task-agnostic but semantically meaningful features and propose to introduce the fixed 2D feature extractors pre-trained on large-scale datasets to help the training of our LTA-PCS for sampling more semantic meaningful points, where the specific tasks and the corresponding labels are both not required. Specifically, to solve the heterogeneous problem of image and point cloud data, we first project 3D point clouds to multi-view depth maps, which can be readily used by 2D feature extractors to extract semantic meaningful features. Then, we calculate the loss based on both the 3D coordinates and the extracted features and train the sampling network, which can enforce it to recognize meaningful points in terms of both geometric and semantic aspects. After training, we directly use the trained sampling network to generate the simplified point clouds for different downstream tasks, in which an inference alignment strategy is used to ensure the simplified point cloud is a subset of the original dense point cloud.

In the aforementioned training paradigm, another issue is that the projection from point clouds to multi-view depth maps will inevitably cause information loss, causing inaccurate learning of semantic meaningful points. To this end, we propose a new loss function called semantic loss, which includes both intra- and inter-view losses. Specifically, the intra-view loss minimizes the information loss between the dense point cloud and simplified point cloud under the same views, while the inter-view loss is calculated based on the relationship between different views. By introducing the semantic loss, we can consider both the information under the same and cross views, which can help the sampling network to generate more semantic meaningful points.

Moreover, our LTA-PCS framework is also general. Although it is designed for task-agnostic point cloud sampling, we can further combine it with the task-specific loss for task-specific point cloud sampling to achieve better performance when the downstream task is given.

The contributions of LTA-PCS are shown as follows:

- To the best of our knowledge, we propose the first learnable task-agnostic point cloud sampling framework called LTA-PCS, in which both the geometric and semantic information are considered and the task-oriented labels are not required.
- We introduce the pre-trained 2D feature extractor to compute the inter- and intra-view learning objective functions to help train the sampling network in our LTA-PCS framework, which can enforce the sampling network to sample more semantically meaningful points.
- Extensive experiments on multiple downstream tasks demonstrate the effectiveness of our LTA-PCS.

## 2. Related Work

**Deep learning on point clouds.** Recently, deep learning methods [18, 22, 23, 37–39, 48, 53] attract many attentions, in which a large number of methods using point clouds are proposed for 3D scene understanding [9–11, 17, 19–21, 46, 49, 50]. For example, PointNet [27] uses multi-layer perceptron to process points for different tasks. PointNet++ [28] proposes a hierarchical structure to extract features. Li et al. [16] proposed PointCNN to perform convolution operations on point clouds. Wu et al. [45] proposed PointConv to introduce point density when perform convolution on point clouds. There are also many methods on point cloud segmentation [40, 41, 43] and object detection [29, 34, 56]. In addition, approaches are also proposed to focus on point cloud completion [4, 32, 54], point cloud registration [2, 33]. On the other hand, some methods project point clouds into volumetric [24, 57] or multi-view [13, 31, 35, 52] data forms and process them using neural networks. For example, Zhang et al. [55] project point clouds into multi-view depth images and process them using pretrained CLIP model [31]. Goyal et al. [8] propose

Table 1. Comparison between our LTA-PCS framework with existing point cloud sampling methods.

| Methods | Semantic information | Task-agnostic |
|---|---|---|
| FPS | ✗ | ✓ |
| Learning-based methods [5, 7, 15] | ✓ | ✗ |
| LTA-PCS | ✓ | ✓ |

SimpleView projection methods for point cloud classification. Zhu et al. [58] utilize pretrained CLIP model for 3D open-world learning. Although these methods use CNNs or CLIP models for learning point cloud features, they do not focus on point cloud sampling. Besides, we utilize the 2D feature extractor in a different perspective, in which the 2D feature extractors are used to facilitate the training of our LTA-PCS for more semantically meaningful sampling.

**Point cloud sampling methods.** Recently, many point cloud sampling methods were proposed. In Table 1, for the task-agnostic point cloud sampling, the FPS method samples the point that is farthest from the previously sampled ones iteratively. However, this method only uses point coordinates, which does not consider the semantic information of points and leads to a sub-optimal solution when using the sampled points for downstream tasks. Besides, many learning-based approaches were proposed [5, 30, 44]. For example, Dovrat et al. [7] proposed S-NET to use networks for point cloud sampling, where a sampling regularization loss is used to train this S-NET. Lang et al. [15] proposed SampleNet to use differentiable relaxation for point cloud sampling. Qian et al. [30] proposed a task-oriented point cloud downsampling approach. Attention operation is also used for point cloud sampling [42, 51]. Although these methods use networks to sample semantic meaningful points from a dense point cloud, they are task-oriented. Therefore, these approaches need specific downstream tasks to train the sampling network, which is often infeasible in real-world applications. In contrast, our LTA-PCS framework is task-agnostic, and the comparison between our LTA-PCS and the existing methods is shown in Table 1.

# 3. Methodology

## 3.1. Problem Statement

Given a dense point cloud with $n$ points $P = \{p_i \in \mathbb{R}^3, i = 1, \dots, n\}$ and a target size $m \leq n$, our goal in this work is to learn a sampling network that can sample $m$ points from $P$ and generate a simplified point cloud that can best represent the dense point cloud with task-agnostic semantic meaning as follows:

$$T(P) \subset P, \ \ |T(P)| = m \leq n, \tag{1}$$

where $T(\cdot)$ is the operation of the sampling network, and $T(P)$ is the simplified point cloud. In our LTA-PCS, the learning process of the sampling network $T(\cdot)$ is not related to any downstream tasks. In other words, we aim to find the best sampling network to generate the simplified point cloud that can perform well on the downstream tasks. Traditional point cloud sampling methods (e.g., FPS) only sample the points based on 3D coordinates, which lacks semantic information. To introduce semantic information in sampling, learning-based sampling methods like S-NET [7] and SampleNet [15] use the task-specific loss to train the sampling network. However, when the task is not given, it is still an open problem on how to introduce such a semantic-aware loss for guiding the sampling network to sample semantically meaningful points. Therefore, we propose to use effective and well-performed pretrained 2D feature extractors to facilitate the training of the sampling network for LTA-PCS in a task-agnostic way.

## 3.2. Overview

Fig. 2 shows the overview when training our LTA-PCS framework. Given a dense point cloud, we use the sampling network in LTA-PCS to sample points from this point cloud and generate the simplified point cloud. To utilize the pretrained 2D feature extractors, we first project both the dense and simplified point clouds to multi-view depth maps. Then, we use the pretrained 2D feature extractor to extract semantically meaningful features based on multi-view depth maps. In our implementation, we choose the vision branch of the CLIP model [31] as our pretrained 2D feature extractor, and the parameters of the 2D feature extractor are frozen in the training process. After that, we calculate the proposed semantic loss including both inter- and intra-view losses based on the output feature from 2D feature extractors, which introduces the semantic-level supervision to help the sampling network in our LTA-PCS framework and sample more semantically meaningful points. Besides, to fully utilize the 3D coordinate information of point clouds, we also use geometric loss [7] to introduce geometric-level supervision when training our LTA-PCS framework in the training process. We update the sampling network based on both geometric loss and semantic loss.

**Sampling network.** Our sampling model follows the architecture of [7, 27]. The input points are processed by a series of $1 \times 1$ convolution layers, which produce a feature vector for each point. Then, a symmetric feature-wise max pooling operation is applied to obtain a global feature vector. Finally, we use several fully-connected layers. The output of the last layer is the set of generated points.

**Projection.** Directly using pretrained 2D feature extractor is infeasible because of the heterogeneous problem between 2D images and 3D point clouds. To solve this problem, we need to project point clouds to multi-view

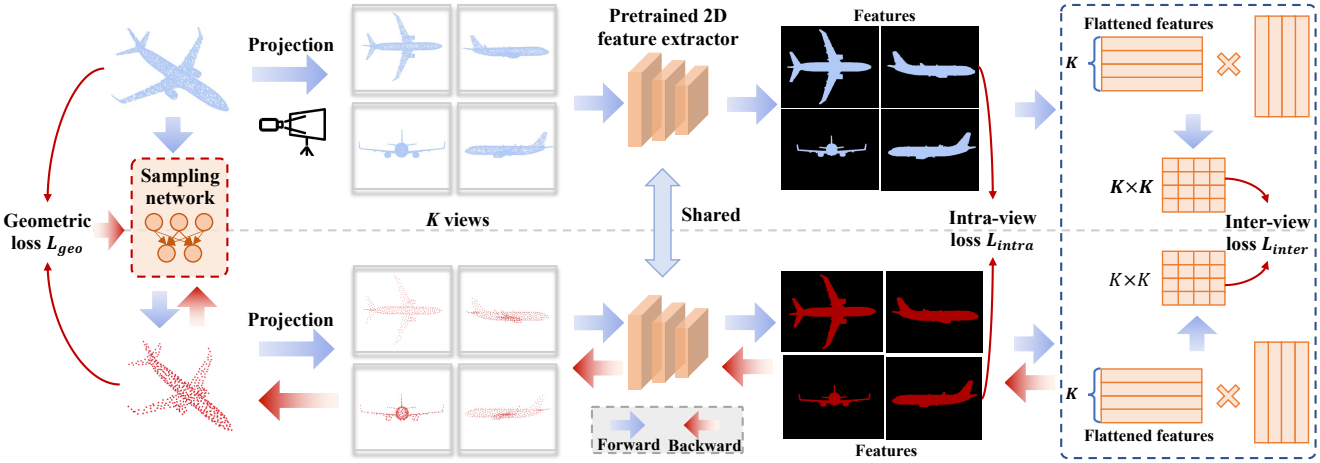


Figure 2. Training procedure of our LTA-PCS framework. Given a dense point cloud, we first use the initialized sampling network to sample points from it and generate a simplified point cloud. Then, we project both the dense and simplified point clouds to multi-view depth maps, which will be used for feature extraction by a frozen pretrained 2D feature extractor. We calculate the semantic loss including intra-view and inter-view losses using the extracted features. Geometric loss is also used in training. At the inference stage, we directly use the trained sampling network to sample points from dense point clouds.

depth maps. Specifically, inspired by [55], we take the bottom view as an example, the point with the coordinate of $(x, y, z)$ will be projected to $(\lceil x/z \rceil, \lceil y/z \rceil)$. Also, as the vision branch of CLIP model aims to process images with three channels, we copy the projected depth maps three times and form the input of the CLIP model, which ensures there is no shape mismatch when using the CLIP model.

**Inference.** At inference, we use the sampling network in Fig. 2 to sample points from the dense point cloud, which can be used for multiple downstream tasks.

### 3.3. Loss Function

**Overall loss function.** Formally, the overall loss function of our LTA-PCS can be written as follows:

$$L = L_{geo} + \alpha L_{sem}, \tag{2}$$

where $L_{geo}$ is the geometric loss, and $L_{sem}$ is the newly proposed semantic loss, which includes both the intra- and inter-view losses. $\alpha$ is the coefficient to balance these two terms. In LTA-PCS, the geometric and semantic losses aim to preserve the geometric and semantic consistency between the original point cloud and the simplified point cloud.

**Geometric loss.** We denote $Q$ as the simplified point cloud, i.e., $Q = T(P)$. Inspired by [7], the geometric loss is defined as follows:

$$\begin{aligned} L_{geo} &= L_a(Q,P) + \beta L_w(Q,P) + \gamma L_s(Q,P), \\ \text{where } L_a(Q,P) &= \frac{1}{|Q|}\sum_{q\in Q} \min_{p\in P} ||q-p||_2^2, \\ L_w(Q,P) &= \max_{q\in Q} \min_{p\in P} ||q-p||_2^2, \\ L_s(Q,P) &= \frac{1}{|P|}\sum_{p\in P} \min_{q\in Q} ||p-q||_2^2. \end{aligned} \tag{3}$$

Here, $L_a(Q,P)$ is to minimize the average distance between $Q$ and $P$. $L_w(Q,P)$ is to minimize the maximum distance between $Q$ and $P$. $L_s(Q,P)$ ensures the points in $Q$ evenly spread in the 3D coordinate space. Following [7], $\beta$ and $\gamma$ are the loss weights.

**Semantic loss.** To enforce the sampling network to sample more semantically meaningful points, we also introduce the newly proposed semantic loss $L_{sem}$ in the training process. Formally, let us suppose the dense point cloud as $P$ and the simplified point cloud generated by the initialized sampling network as $Q$. The semantic loss consists of intra-view and inter-view loss, which can be written as follows:

$$L_{sem} = L_{intra} + \lambda L_{inter}. \tag{4}$$

The intra-view loss aims to minimize the semantic information loss under the same view as follows:

$$L_{intra} = \sum_{k=1}^{K} MSE[E(J_k(P)), E(J_k(Q))], \tag{5}$$

**Algorithm 1** Training procedure of our LTA-PCS.

**Input:** Dense point cloud $P$; Simplified point cloud $Q$; Randomly initialized sampling network $T$; Number of projected views $K$; Pretrained 2D feature extractor $E$;

**for** each iteration in the training process **do**
  // Calculate geometric loss $L_{geo}$
  Use $T$ to generate the simplified point cloud $Q$;
  Calculate geometric loss $L_{geo}$ based on dense point cloud $P$ and the simplified point cloud $Q$ using Eq. (3);
  // Calculate semantic loss $L_{sem}$
  **for** view $k$ in $K$ **do**
    Project dense point cloud $P$ and simplified point cloud $Q$ to the $k$-th view depth map $J_k(P)$ and $J_k(Q)$, respectively;
    Extract the features of $J_k(P)$ and $J_k(Q)$ using the 2D feature extractor $E$, respectively;
  **end for**
  Calculate intra- and intra-view losses by Eq. (5) and Eq. (6);
  Calculate semantic loss $L_{sem}$ based on Eq. (4);
  Calculate overall object function $L$ using Eq. (2);
  Back propagate the gradients and update the parameters of the sampling network $T$;
**end for**

**Output:** The optimized sampling network $T$;

where $J_k(\cdot)$ denotes the projection operation for the $k$-th view, and $E(\cdot)$ is the 2D feature extractor for extracting semantic features. $MSE(\cdot)$ is the mean square error, and $K$ is the total number of projected views.

As the projection operation inevitably causes information loss and the relationship of different views can also provide essential information [25, 26], we calculate the inter-view loss for compensation as follows:

$$\begin{aligned} L_{inter} &= MSE(\mathbf{\Phi}\mathbf{\Phi}^T, \mathbf{\Psi}\mathbf{\Psi}^T), \\ \text{where } \mathbf{\Phi} &= \underset{K}{Concat}\left[Flat(E(J_k(P)))\right], \\ \mathbf{\Psi} &= \underset{K}{Concat}\left[Flat(E(J_k(Q)))\right]. \end{aligned} \tag{6}$$

Here, $Flat(\cdot)$ is the flatten operation. $Concat_K$ concatenate the vectors along different views. By using the inter-view loss, we obtain the relation information in different views from dense point cloud and enforce the simplified point cloud to mimic this information.

Based on the intra-view loss $L_{intra}$, inter-view loss $L_{inter}$, and the geometric loss $L_{geo}$, we obtain the loss function $L$ in Eq. (2) to train the sampling network $T$, which is used to generate the simplified point cloud at inference. The training procedure is also illustrated in Alg. 1.

### 3.4. Inference

At inference, we use the trained sampling network $T$ to generate the coordinates of the sampled points. However, as we use several fully-connected layers to directly predict the coordinates of the sampled points at the end of the sampling network, we cannot ensure the simplified point cloud $Q$ generated by the sampling network to be a subset of dense point cloud $P$. Thus, following S-Net [7], we can obtain the points in $Q$ to its nearest neighbors in dense point cloud $P$. Specifically, for each point $q \in \mathbb{R}^3$ in $Q$, the matched point $q^*$ is written as follows:

$$q^* = \underset{p \in P}{argmin} \, ||q - p||_2^2. \tag{7}$$

After the matching process for $Q$ with $m$ points, we can obtain the simplified point cloud $Q^* = \{q_i^*\}_{i=1}^m$, which is a subset of $P$. Note that the matching process is very fast.

### 3.5. Downstream Tasks Usage

After training the sampling network in our LTA-PCS, we can use it to effectively generate the simplified point cloud with the information from both 3D coordinates and the semantic meanings. As the downstream tasks are not decided at this stage, we store these simplified point clouds as a new point cloud dataset to simulate this scenario. When the downstream tasks are given, we use the simplified point clouds to train different backbone network architectures (e.g., PointNet [27]) for different downstream tasks as before, and the backbone network is initialized from scratch.

### 3.6. Discussion

**Necessity of Learnable Task-Agnostic Point Cloud Sampling.** We would emphasize that LTA-PCS tries to solve the point cloud sampling under an important scenario where the collected point clouds are given but the downstream tasks are not given. In real-world applications, this scenario often occurs. We often use LiDAR to collect point clouds and store them. When specific tasks are given, we use the stored point cloud data for downstream tasks. Thus, sampling semantically meaningful points from dense collected point clouds becomes critical for storage saving. Existing learning-based methods [15] use task-specific loss to introduce semantic information in sampling, which is infeasible in this scenario as the downstream tasks and the corresponding labels are required. Although we can remove task-specific loss in these learning-based methods for task-agnostic point cloud sampling, semantic information will not be considered in this case. Moreover, the task-specific loss will degrade the generalization ability of these methods on different datasets. LTA-PCS targets at solving the point cloud sampling under a new setting, which cannot be achieved by existing learning-based approaches.

**Discussion on the reason that our task-agnostic point cloud sampling method (i.e., LTA-PCS) yields better results relative to task-specific methods (e.g., S-Net, SampleNet).** We hypothesize this is because we introduce the 2D pretrained model in the training process. Therefore, the knowledge in 2D pretrained model can significantly improve the performance of the sampling network. In contrast, due to limited network capacity or dataset size, there is no general and well-performed 3D pretrained model when compared with 2D pretrained models. Therefore, although existing state-of-the-art point cloud sampling methods are task-intensive, they cannot well utilize the prior knowledge in 2D pretrained model, resulting worse performance than our task-agnostic method LTA-PCS. Our LTA-PCS framework can be also combined with task-specific loss, and the results show the performance can be further boosted.

**Discussion on the necessity of 2D feature extractor.** We choose the pretrained 2D feature extractor to introduce semantic information when training our LTA-PCS framework as there is no existing well-performed pretrained 3D feature extractor on large-scale datasets in the literature. Current 3D feature extractors like PointNet [27] or PointNet++ [28] are trained based on specific tasks and datasets, which cannot be used in the task-agnostic setting. We hypothesize that this is because of the lack of large-scale high-quality 3D point cloud datasets. Therefore, although we encounter information loss in the projection process, we choose a 2D feature extractor to introduce semantic information when constructing our LTA-PCS.

# 4. Experiments

## 4.1. Implementation Details

We follow [7] to utilize a set of multi-layer perceptrons (MLPs) as the sampling network, and batch normalization and ReLU layers are appended after each MLP. In training, we train the sampling network for 50 epochs with Adam optimizer [14] on both ModelNet40, ShapeNet core55 and ScanObjectNN datasets. The batch size is set as 8 and the learning rate is $5e^{-4}$. We set the weight decay as $1e^{-4}$. In the training process, we set the hyperparameter $\alpha$ in Eq. (2) as 1. $\beta$ and $\gamma$ in Eq. (3) are set as 1 and 1, respectively, and $\lambda$ in Eq. (4) is set as 0.01. The number of views (i.e., $K$) is set as 10. After training the sampling network, we use it to generate simplified point clouds. Then, we use the generated simplified point clouds to train an initialized backbone network for different downstream tasks. For the point cloud classification task, we use PointNet [27] as our backbone network and train the initialized PointNet for 200 epochs with a batch size of 12. The learning rate is set as $1e^{-3}$ with weight decay of $1e^{-4}$. Step learning rate decay is used. The learning rate will multiply 0.7 after every 20 epochs. For the point cloud completion and registration tasks, we use PCN [54] and PointNetLK [1] as baseline methods, respectively, and the training epochs are set as 400 and 200, respectively.

## 4.2. Point Cloud Classification

In Table 2, we compare our LTA-PCS with several existing point cloud sampling methods on the ModelNet40 [47], ShapeNet [3] and ScanObjectNN (hardest perturbed variant (PB_T50_RS)) [36] for point cloud classification, where the instance accuracy is used as the evaluation metric. Specifically, Random and FPS are task-agnostic non-learnable methods, and we directly sample a specific number of points by using the corresponding methods. The baseline approaches S-Net [7], SampleNet [15] are task-specific learnable methods, and we first train the sampling network and use the trained network to downsample the input point cloud to a predefined number of points. However, we can remove the task-specific loss in these learnable sampling approaches to achieve task-agnostic point cloud sampling, which are denoted by S-Net* and SampleNet*, respectively. In contrast, LTA-PCS is a learnable approach without task-specific loss in training. Therefore, it is a task-agnostic point cloud sampling method. In Table 2, we observe that our LTA-PCS achieves better results than other methods. Remarkably, we outperform the state-of-the-art SampleNet* under the task-agnostic setting by 2.1% when sampling to 256 points on ModelNet40. Moreover, LTA-PCS even performs better than the task-specific learnable point cloud sampling methods S-Net and SampleNet. One possible explanation is that S-Net and SampleNet use pretrained 3D feature extractors (e.g., PointNet [27]) to calculate the task loss, and their results are limited by the capacity of 3D feature extractors.

## 4.3. Point Cloud Retrieval

We also follow S-Net [7] to report the results of the point cloud retrieval task. For point cloud retrieval, we use ModelNet40 to evaluate our LTA-PCS framework, where the mean average precision (mAP) evaluation metric is used, In Table 3, our LTA-PCS achieves promising performance. Specifically, we first downsample the point cloud to the corresponding number of points, and then extract the features by PointNet. Following [7], the results are computed by $L_2$ distance of the shape descriptor, which is the feature of the penultimate layer.

## 4.4. Point Cloud Completion

We follow PCN [54] to reconstruct the simplified point clouds into the original dense point clouds with 2,048 points on ShapeNet Core55, and we provide the $L_1$ Chamfer Distance (CD) results of different methods in Fig. 3. Note that for the task-specific methods, we need to additionally train the sampling network when training the PCN method. From

Table 2. Comparison of our LTA-PCS with other methods under the point cloud classification task. * indicates we remove the task-specific loss in training for task-agnostic setting. "Oracle" denotes the accuracy using original dense point clouds.

| Datasets | | ModelNet (Oracle: 90.3) | | | | ShapeNet (Oracle: 85.0) | | | |
|---|---|---|---|---|---|---|---|---|---|
| #Sampled points | | 64 | 128 | 256 | 512 | 64 | 128 | 256 | 512 |
| Task-Agnostic | Random | 81.5 | 86.2 | 86.6 | 88.2 | 79.3 | 80.9 | 81.9 | 83.2 |
| | FPS | 86.1 | 87.9 | 88.1 | 88.3 | 80.5 | 81.5 | 82.4 | 83.3 |
| | S-Net* [7] | 86.1 | 87.8 | 88.2 | 88.5 | 80.6 | 81.3 | 82.2 | 83.0 |
| | SampleNet* [15] | 86.6 | 88.3 | 87.9 | 88.4 | 80.3 | 81.6 | 82.4 | 82.9 |
| | **LTA-PCS** | **88.2** | **89.1** | **90.0** | **90.3** | **82.0** | **83.1** | **83.5** | **84.4** |
| Task-Specific | S-Net [7] | 87.7 | 88.1 | 88.4 | 89.0 | 80.9 | 82.3 | 82.7 | 83.6 |
| | SampleNet [15] | 87.9 | 88.3 | 88.0 | 88.7 | 81.0 | 82.5 | 82.7 | 83.4 |

Table 3. Comparison of our LTA-PCS with other methods under the point cloud retrieval task. * indicates we remove the task-specific loss in training for task-agnostic setting.

| #Sampled points | 64 | 128 | 256 | 512 |
|---|---|---|---|---|
| Task-Agnostic | | | | |
| Random | 66.1 | 69.4 | 72.5 | 73.3 |
| FPS | 69.1 | 72.5 | 73.0 | 74.2 |
| S-Net* [7] | 69.4 | 71.7 | 73.6 | 74.1 |
| SampleNet* [15] | 70.4 | 72.3 | 72.6 | 74.7 |
| **LTA-PCS** | **71.4** | **73.2** | **74.8** | **77.2** |
| Task-Specific | | | | |
| S-Net [7] | 70.5 | 72.7 | 74.0 | 76.2 |
| SampleNet [15] | 70.9 | 72.5 | 73.6 | 76.0 |

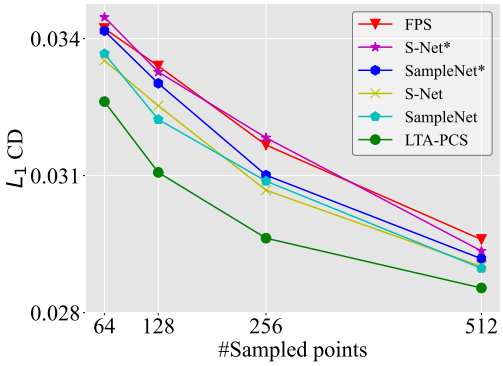


Figure 3. Performance on the point cloud completion task. Lower $L_1$ Chamber Distance ($L_1$ CD) denotes better performance.

Fig. 3, we observe that the quantitative results of our LTA-PCS are better than other methods under different numbers of sampled points. The results show our LTA-PCS is effective for different downstream tasks.

**Visualization.** We provide the visualization results of different methods in Fig. 4. Specifically, in Fig. 4(a), the simplified point clouds are produced by FPS, S-Net and LTA-PCS, where the number of points in each simplified point cloud are 64, and in Fig. 4(b), we observe that the reconstruction quality of LTA-PCS is better than other methods.

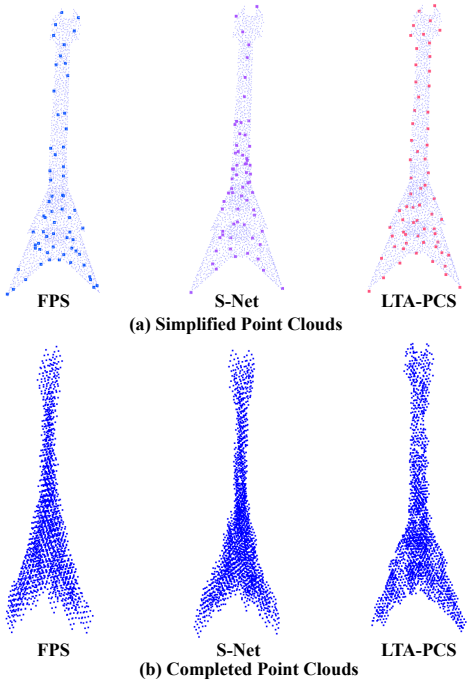


Figure 4. (a). Simplified point clouds. (The sampled points are enlarged for better visualization.) (b). Completed point clouds by using the corresponding simplified point clouds as input.

### 4.5. Ablation Study

**Effect of different losses.** In Table 4 we provide three alternative variants of our LTA-PCS: LTA-PCS (w/o inter & w/o intra), LTA-PCS (w/o inter), LTA-PCS (w/o intra). Specifically, for LTA-PCS (w/o inter & w/o intra), we remove the

Table 4. Ablation on using different losses.

| #Sampled points | 128 | 256 |
|---|---|---|
| LTA-PCS | **89.1** | **90.0** |
| LTA-PCS (w/o inter & w/o intra) | 87.8 | 88.2 |
| LTA-PCS (w/o inter) | 88.6 | 89.4 |
| LTA-PCS (w/o intra) | 88.3 | 89.1 |

Table 5. Ablation on using different feature extractors.

| #Sampled points | 128 |
|---|---|
| LTA-PCS (PointNet) | 88.0 |
| LTA-PCS (R50 trained on ImageNet) | 88.3 |
| LTA-PCS (R50 of CLIP vision encoder) | 89.1 |

Table 6. Ablation on using different numbers of views.

| # views | 2 | 6 | 10 |
|---|---|---|---|
| LTA-PCS | 88.2 | 88.9 | 89.1 |

semantic losses (i.e., both inter-view and intra-view losses), which is the same as the S-Net [7] without using the task loss. For LTA-PCS (w/o inter), we only use the intra-view loss without using the inter-view loss to train our LTA-PCS framework. For LTA-LTA-PCS (w/o intra), we only use the inter-view loss without using the intra-view loss to train the LTA-PCS framework. In Table 4, our LTA-PCS is better than these three alternative variants, which shows that it is beneficial to utilize both intra-view and inter-view losses.

**Effect of 2D feature extractor.** In Table 5, we compare the results of our LTA-PCS by using different pretrained vision backbones [6, 12] to show the effect of 2D feature extractor. Specifically, for LTA-PCS (PointNet), we directly use the PointNet backbone to extract the features of the original dense point cloud and simplified point cloud and calculate the MSE distance of corresponding features to preserve semantic consistency. For LTA-PCS (R50 trained on ImageNet), we use the ResNet-50 [12] model trained on ImageNet as the feature extractor. For LTA-PCS (R50 of CLIP vision encoder) and LTA-PCS (VIT-B/16 of CLIP vision encoder), we use the pretrained CLIP vision encoder with different backbones. In Table 5, first, the results of LTA-PCS (PointNet) are lower than other methods a lot, which indicates that the 3D feature extractor (i.e., PointNet) cannot provide high-quality semantic supervision to train the sampling network. Second, we observe that the results of LTA-PCS (R50 of CLIP vision encoder) are better than the results of LTA-PCS (R50 trained on ImageNet) by a large margin, which shows the capacity of the feature extractor is critical when training the sampling network. Therefore, we choose the pre-trained 2D feature extractor to maintain semantic consistency when training the sampling network instead of using the pre-trained 3D feature extractor.

**Effect of the number of views.** In Table 6, we report the performance results of using 128 sampled points as input to analyze the effect of the number of projected views, where we report the results on the ModelNet40 dataset based on the PointNet backbone. When the number of views increases from 2 to 10, LTA-PCS can achieve better performance results, which indicates that it is effective to use more views for maintaining the semantic consistency between the original and simplified point clouds. Therefore, we set the projected number of views as 10 to achieve better performance with acceptable memory usage.

### 4.6. Further Analysis

Table 7. Inference time and GPU memory usage of our LTA-PCS with different sampled points.

| #Sampled points | Inference time (ms) | GPU usage (MB) |
|---|---|---|
| 64 | 0.28 | 578 |
| 128 | 0.37 | 616 |
| 256 | 0.58 | 724 |
| 512 | 1.05 | 940 |
| 2048 | 1.77 | 2326 |

**Comparison on inference time and GPU memory usage.** In Table 7, we take the ModelNet40 dataset as an example and compare the inference time and GPU memory usage under different sampled points when using PointNet as the backbone network. As shown in Table 7, we observe that using fewer points can reduce inference time and GPU memory usage greatly, which further demonstrates the advantages of point cloud sampling.

## 5. Conclusion

In this work, we propose the Learnable Task-Agnostic Point Cloud Sampling (LTA-PCS) framework for task-agnostic point cloud sampling task, which aims to preserve the geometric and semantic consistency in point cloud sampling. Specifically, we first propose to use the pre-trained 2D feature extractor for training the sampling network, and use semantic loss including both inter-view and intra-view losses. Comprehensive experimental results on multiple downstream tasks show the effectiveness of LTA-PCS. In the future, we will continue to explore more tasks based on our LTA-PCS method.

**Acknowledgement.** This work is supported by National Science and Technology Major Project (2022ZD0116405) and National Natural Science Foundation of China (No. 61932002, No. 62306025, No. 92367204).

## References


[1] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6

[2] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019. 2

[3] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 6

[4] Xuelin Chen, Baoquan Chen, and Niloy J Mitra. Unpaired point cloud completion on real scans using adversarial training. *arXiv preprint arXiv:1904.00069*, 2019. 2

[5] Ta-Ying Cheng, Qingyong Hu, Qian Xie, Niki Trigoni, and Andrew Markham. Meta-sampler: Almost-universal yet task-oriented sampling for point clouds. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, page 694–710, Berlin, Heidelberg, 2022. Springer-Verlag. 3

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 8

[7] Oren Dovrat, Itai Lang, and Shai Avidan. Learning to sample. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 3, 4, 5, 6, 7, 8

[8] Ankit Goyal, Hei Law, Bowei Liu, Alejandro Newell, and Jia Deng. Revisiting point cloud shape classification with a simple and effective baseline. In *International Conference on Machine Learning*. PMLR, 2021. 2

[9] Jinyang Guo, Jiaheng Liu, and Dong Xu. Jointpruning: Pruning networks along multiple dimensions for efficient point cloud processing. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021. 2

[10] Jinyang Guo, Jiaheng Liu, and Dong Xu. 3d-pruning: A model compression framework for efficient 3d action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(12):8717–8729, 2022.

[11] Jinyang Guo, Dong Xu, and Wanli Ouyang. Multidimensional pruning and its extension: A unified framework for model compression. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 8

[13] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 2

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 6

[15] Itai Lang, Asaf Manor, and Shai Avidan. Samplenet: Differentiable point cloud sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1, 3, 5, 6, 7

[16] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *NeruIPS*, 2018. 2

[17] Jiaheng Liu and Dong Xu. Geometrymotion-net: A strong two-stream baseline for 3d action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 31 (12):4711–4721, 2021. 2

[18] Jiaheng Liu, Yudong Wu, Yichao Wu, Chuming Li, Xiaolin Hu, Ding Liang, and Mengyu Wang. Dam: discrepancy alignment metric for face recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3814–3823, 2021. 2

[19] Jiaheng Liu, Jinyang Guo, and Dong Xu. Apsnet: Towards adaptive point sampling for efficient 3d action recognition. *IEEE Transactions on Image Processing*, 2022. 2

[20] Jiaheng Liu, Jinyang Guo, and Dong Xu. Geometrymotion-transformer: An end-to-end framework for 3d action recognition. *IEEE Transactions on Multimedia*, 2022.

[21] Jiaheng Liu, Tong He, Honghui Yang, Rui Su, Jiayi Tian, Junran Wu, Hongcheng Guo, Ke Xu, and Wanli Ouyang. 3d-queryis: A query-based framework for 3d instance segmentation. *arXiv preprint arXiv:2211.09375*, 2022. 2

[22] Jiaheng Liu, Haoyu Qin, Yichao Wu, Jinyang Guo, Ding Liang, and Ke Xu. Coupleface: Relation matters for face recognition distillation. In *European Conference on Computer Vision*, pages 683–700. Springer, 2022. 2

[23] Jiaheng Liu, Zhipeng Yu, Haoyu Qin, Yichao Wu, Ding Liang, Gangming Zhao, and Ke Xu. Oneface: one threshold for all. In *European Conference on Computer Vision*, pages 545–561. Springer, 2022. 2

[24] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015. 2

[25] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019. 5

[26] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *ICCV*, pages 5007–5016, 2019. 5

[27] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification

and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017. 2, 3, 5, 6

[28] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2, 6

[29] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *ICCV*, 2019. 2

[30] Yue Qian, Junhui Hou, Qijian Zhang, Yiming Zeng, Sam Kwong, and Ying He. Mops-net: A matrix optimization-driven network fortask-oriented 3d point cloud downsampling. *arXiv preprint arXiv:2005.00383*, 2020. 3

[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 2021. 2, 3

[32] Muhammad Sarmad, Hyunjoo Jenny Lee, and Young Min Kim. Rl-gan-net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 2

[33] Vinit Sarode, Xueqian Li, Hunter Goforth, Yasuhiro Aoki, Rangaprasad Arun Srivatsan, Simon Lucey, and Howie Choset. Pcrnet: Point cloud registration network using pointnet encoding. *arXiv preprint arXiv:1908.07906*, 2019. 2

[34] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 2

[35] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, 2015. 2

[36] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *International Conference on Computer Vision (ICCV)*, 2019. 6

[37] Haodi Wang, Kai Dong, Zhilei Zhu, Haotong Qin, Aishan Liu, Xiaolin Fang, Jiakai Wang, and Xianglong Liu. Transferable multimodal attack on vision-language pre-training models. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 102–102. IEEE Computer Society, 2024. 2

[38] Jiakai Wang, Aishan Liu, Zixin Yin, Shunchang Liu, Shiyu Tang, and Xianglong Liu. Dual attention suppression attack: Generate adversarial camouflage in physical world. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 8565–8574. Computer Vision Foundation / IEEE, 2021.

[39] Jiakai Wang, Aishan Liu, Xiao Bai, and Xianglong Liu. Universal adversarial patch attack for automatic checkout using perceptual and attentional bias. *IEEE Trans. Image Process.*, 31:598–611, 2022. 2

[40] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 2

[41] Xinlong Wang, Shu Liu, Xiaoyong Shen, Chunhua Shen, and Jiaya Jia. Associatively segmenting instances and semantics in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 2

[42] Xu Wang, Yi Jin, Yigang Cen, Tao Wang, Bowen Tang, and Yidong Li. Lightn: Light-weight transformer network for performance-overhead tradeoff in point cloud downsampling. *arXiv preprint arXiv:2202.06263*, 2022. 3

[43] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 38(5):1–12, 2019. 2

[44] C. Wen, B. Yu, and D. Tao. Learnable skeleton-aware 3d point cloud sampling. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17671–17681, Los Alamitos, CA, USA, 2023. IEEE Computer Society. 3

[45] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, 2019. 2

[46] Yue Wu, Xidao Hu, Yue Zhang, Maoguo Gong, Wenping Ma, and Qiguang Miao. Sacf-net: Skip-attention based correspondence filtering network for point cloud registration. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(8):3585–3595, 2023. 2

[47] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 6

[48] Ge Yang, Chao Zhang, Ling Gao, Yufei Guo, and Jinyang Guo. Domain adaptive channel pruning. *Electronics*, 13(5), 2024. 2

[49] Honghui Yang, Tong He, Jiaheng Liu, Hua Chen, Boxi Wu, Binbin Lin, Xiaofei He, and Wanli Ouyang. Gd-mae: generative decoder for mae pre-training on lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9403–9414, 2023. 2

[50] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3323–3332, 2019. 2

[51] Yang Ye, Xiulong Yang, and Shihao Ji. Apsnet: Attention based point cloud sampling. *arXiv preprint arXiv:2210.05638*, 2022. 3

[52] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 2

[53] Zhipeng Yu, Jiaheng Liu, Haoyu Qin, Yichao Wu, Kun Hu, Jiayi Tian, and Ding Liang. Icd-face: Intra-class compactness distillation for face recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21042–21052, 2023. 2

[54] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 international conference on 3D vision (3DV)*. IEEE, 2018. 2, 6

[55] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2, 4

[56] Lichen Zhao, Jinyang Guo, Dong Xu, and Lu Sheng. Transformer3d-det: Improving 3d object detection by vote refinement. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021. 2

[57] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 2

[58] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyao Zeng, Shanghang Zhang, and Peng Gao. Pointclip v2: Adapting clip for powerful 3d open-world learning. *arXiv preprint arXiv:2211.11682*, 2022. 3