

# Resolution-free Point Cloud Sampling Network with Data Distillation

Tianxin Huang<sup>1\*</sup>, Jiangning Zhang<sup>1,2\*</sup>, Jun Chen<sup>1</sup>, Yuang Liu<sup>1</sup>, and Yong Liu<sup>1,3†</sup>

<sup>1</sup> APRIL Lab, Zhejiang University, HangZhou, China

<sup>2</sup> Tencent Youtu Laboratory

<sup>3</sup> Huzhou Institute of Zhejiang University  
yongliu@iipc.zju.edu.cn

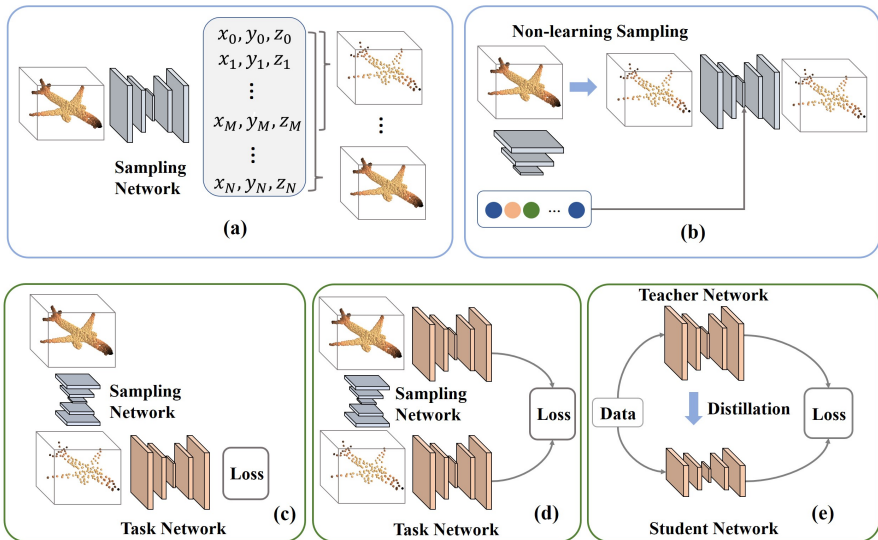
**Abstract.** Down-sampling algorithms are adopted to simplify the point clouds and save the computation cost on subsequent tasks. Existing learning-based sampling methods often need to train a big sampling network to support sampling under different resolutions, which must generate sampled points with the costly maximum resolution even if only low-resolution points need to be sampled. In this work, we propose a novel resolution-free point clouds sampling network to directly sample the original point cloud to different resolutions, which is conducted by optimizing non-learning-based initial sampled points to better positions. Besides, we first introduce data distillation to assist the training process by considering the differences between task network outputs from original point clouds and sampled points. Experiments on point cloud reconstruction and recognition tasks demonstrate that our method can achieve SOTA performances with lower time and memory cost than existing learning-based sampling strategies.

**Keywords:** 3D point clouds; Down sampling; Resolution free;

## 1 Introduction

With the rapid development of 3D related tasks such as SLAM [3] and recognition [23, 24, 28], point cloud has attracted more and more attentions in computer vision and robots. However, large point clouds limit the efficiency of related algorithms and bring up higher device requirements. Sampling original point clouds to lower resolutions might be an alternative solution to reduce the computational cost on subsequent tasks. Existing works [24, 13, 22, 33, 18, 14] use random sampling or Farthest Point Sampling (FPS) to down sample the point clouds. However, non-learning-based sampling strategies are lack of relevance to down-stream tasks, which blocks their further improvements on task-oriented performances.

In this condition, some researchers [16, 8] propose learning-based sampling methods to improve the task-oriented performances through optimization. S-Net [8] generates initial sampled points directly by fully-connected networks and project them back to their nearest neighbors in original point clouds. Sampled points of [8] has relatively weak relevance with the original points because the projection process from generated sampled points to original point clouds is independent from training, while SamNet [16] gets over this problem by designing a differentiable approximation for projection during



**Fig. 1.** (a) and (b) show differences between progressive implements of existing sampling networks and our resolution-free network, while (c), (d) and (e) show the architecture of former sampling networks, Ours and knowledge distillation.  $N$  and  $M$  denote the number of original and sampled points, respectively.

training. To support different resolutions with a single network, [8, 16] propose progressive implements by generating candidate points with the same number of input point clouds and directly select sampled points from all candidates in order, as illustrated in Fig. 1-(a). Points of multiple sampling resolutions are fed into the task network together to provide constraints for different resolutions. The progress implement always needs to generate sampled points of maximum resolutions even if only low-resolution points are sampled, which introduces extra computational cost. Besides, both S-Net [8] and Sam-Net [16] only use pre-trained task networks to transfer gradients from task-oriented losses in a straightforward process as shown in Fig. 1-(c), which ignore learned distributions of the task network on original point clouds. In other words, they do not make use of the learned knowledge included in pre-trained networks.

In this work, we propose a simple but effective resolution-free Point Cloud sampling network with data Distillation (PCDNet). As presented in Fig. 1-(b), by driving existing initial sampled points from non-learning-based sampling strategies to better positions instead of generating points directly, PCDNet can get over the limitation of fully-connected networks and acquire sampled points of any resolution directly, which avoids the extra computational cost and improves sampling efficiency.

During the training process of PCDNet, we propose a dynamic resolution selection strategy to choose the optimized resolution in each iteration by considering the convergence on each sampling resolution. Resolutions whose task-oriented losses vary larger will acquire higher resolution probabilities to help them converge steadily. Besides, we propose data distillation to assist the optimization by considering differences between task network outputs from original and sampled points, which actually takes advantage

of the knowledge in pre-trained task networks as knowledge distillation methods [12, 25]. As indicated in Fig. 1-(d) and (e), knowledge distillation constrains different networks based on same data, while data distillation constrains different data with same task networks. With the help of data distillation, our sampling network can converge to better results than straightforward training.

Our contributions can be summarized as follows:

- We propose a novel resolution-free point cloud sampling network to sample original point clouds to any resolution directly, where a dynamic resolution selection strategy is proposed to assist the optimization;
- We introduce data distillation by considering differences between the task network outputs from original point clouds and sampled points to supervise the training of sampling network;
- Experiments on reconstruction and recognition tasks demonstrate that PCDNet can outperform former point cloud sampling methods on task-oriented performances with lower time and memory cost than existing learning-based sampling networks.

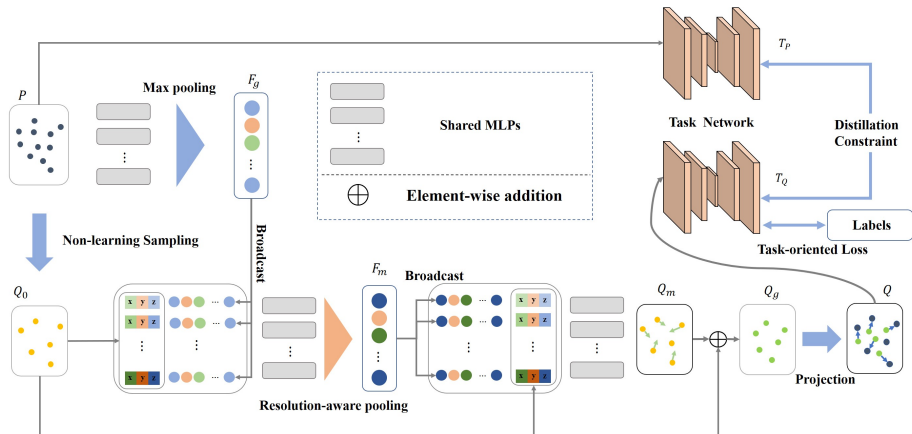
## 2 Related Works

### 2.1 Point Cloud Sampling

Random sampling and farthest point sampling (FPS) [24, 22, 32, 13] are two widely used point cloud sampling strategies. FPS keeps a sampled point set and cyclically adds the point farthest from the sampled set in the remained parts to the sampled set, while random sampling directly down-samples the points by random selection. Since the development of deep learning based methods on point clouds [17, 20, 26, 28], some learning-based works [16, 8] have also been released to enhance the performances of sampled points for specified tasks such as recognition [23, 24, 29, 19, 27] and reconstruction [1, 9, 31], which can greatly outperform non-learning sampling strategies. However, the learning-based works are still quite limited by the network designations based on fully-connected networks. Both of [8] and [16] require the progressive implement to get a single sampling network for all resolutions in one training process, it actually trains a big sampling network contains all possible resolutions, which always generates sampled points of maximum resolutions even if only low-resolution points are sampled and introduces extra computational cost.

### 2.2 Knowledge Distillation

Knowledge Distillation includes a series of methods aiming to train a small student network with the help of a big teacher network. Hinton et al [12] first propose the vanilla knowledge distillation by introducing extra constraints between the outputs of teacher network and student network, which acquires surprising performances on MNIST classification and Automatic Speech recognition. Later works [25, 11, 2, 34, 15] further take advantage of the relations between intermediate features to train deeper and thinner student networks. [5] propose an available distillation framework for object detection, which is a common regression problem. [21, 6, 10] propose data-free methods to reduce



**Fig. 2.** The whole pipeline of PCDNet. Initial seeds  $Q_0$  are sampled from input set  $P$  with a non-learning-based sampling strategy, which is farthest point sampling in this work. Global features  $F_g$  is extracted from  $P$  and merged with  $Q_0$  to construct merged features  $F_m$ , which is further used to predict a displacement field  $Q_m$  including an offset for each point in  $Q_0$  and drive them to better positions  $Q_g$ .  $Q_g$  is projected back to  $P$  to get final sampled points  $Q$ .

the reliance on original training data and improve distillation performances by introducing generated synthetic data. In this work, we propose data distillation to assist the point cloud sampling process based on knowledge distillation methods.

### 3 Methodology

#### 3.1 Resolution-free sampling network

**Network structure.** The whole pipeline of our work is presented in Fig. 2. Input points are sampled with non-learning-based sampling strategies such as FPS [24] to produce initial sampled points, named seeds. Then we aggregate global features from the input points with a set of parameter-shared multilevel perceptions (MLPs) and pooling following [23]. The global features would contain information of original models, which would be concatenated with initial seeds and combined into merged features including information from both initial seeds and original models with MLPs and pooling. The initial seeds will be concatenated again with merged features and fed into MLPs to predict a displacement field including an offset for each initial seed. By moving initial seeds with predicted displacement field, we can acquire generated sampled points of any resolution easily. Finally, we follow [16] to project the generated sampled points back to original point clouds and get final sampled points.

**Resolution-aware pooling.** Specially, to design a network which can easily adapt to different sampling resolutions, we propose resolution-aware pooling to aggregate point features. Common pooling operations such as max pooling or average pooling are not sensitive to the point cloud resolution. In other words, point clouds with different

resolutions may have same features when they have identical shapes, which is not distinguishable to sample multiple resolution point clouds. In this condition, we propose resolution-aware pooling to aggregate resolution-related features. Let  $N$  be the number of points in the original point cloud,  $M$  be current sampling resolution,  $F_i$  be  $i$ -th point feature need to be aggregated. The resolution-aware pooling is defined as

$$F_o = \frac{\sum_1^M F_i}{N}, \quad (1)$$

where  $F_o$  is the aggregated feature. We can see that aggregated features will have relatively smaller values when the resolution is low, larger otherwise. The resolution information is then simply introduced to the network, which can improve its adaptability to different sampling resolutions.

### 3.2 Dynamic resolution selection strategy

During the training process of PCDNet, a certain resolution is chosen in each iteration. To help resolution-free sampling network adapt to different sampling resolutions, we propose a dynamic resolution selection strategy to help decide the training resolution in each iteration. The algorithm is presented in Alg. 1. In details, given the resolution updating interval  $n$ , we evaluate and record errors of the sampling network under different resolutions every  $n/2$  iterations, while adjusting selection probabilities each  $n$  iterations according to the two recorded errors. Resolutions whose errors vary larger will get higher selection probabilities to help them converge steadily. With the Dynamic resolution selection strategy, PCDNet can get good and balanced convergence under multiple resolutions.

### 3.3 Data Distillation

Existing learning-based sampling networks feed the sampling results to pre-trained task networks and get optimized in a straightforward process. They do not make full use of the knowledge included in the pre-trained task networks. In this condition, as shown in Fig. 2, we introduce knowledge distillation constraints between task network outputs from original data and sampled results to "distillate" point clouds, which we named data distillation. Data distillation guides to simplify the data based on networks, while knowledge distillation teaches to simplify the network based on the data.

For point cloud reconstruction, a commonly used task-oriented constraint Chamfer Distance(CD) [9] is adopted, which is defined as

$$\mathcal{L}_{CD}(S_1, S_2) = \frac{1}{2} \left( \frac{1}{|S_1|} \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2 + \frac{1}{|S_2|} \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2 \right), \quad (2)$$

where  $S_1$  and  $S_2$  are two point sets. CD is actually the average distance from points in one set to their nearest neighbors in another set. Let  $P$  and  $Q$  be the original point clouds and sampled results, following [7], we can define the data distillation constraint for point cloud reconstruction as

$$\mathcal{L}_{DD}(P, Q, T_P, T_Q) = \begin{cases} \mathcal{L}_{CD}(P, T_Q), & \text{if } \mathcal{L}_{CD}(P, T_Q) + m > \mathcal{L}_{CD}(P, T_P) \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

---

**Algorithm 1** Training with Dynamic resolution selection
 

---

**Input:** data  $X$ , the number of iterations  $iter$ ,  
 Updating interval  $n$ , the number of resolutions  $m$ ,  
**Initialize Error lists:**  
 $errlist_1, errlist_2, \dots, errlist_m = [], [], \dots, []$ ;  
**Initialize Probabilities:**  
 $prob_1, prob_2, \dots, prob_m = \frac{1}{m}, \frac{1}{m}, \dots, \frac{1}{m}$ ;  
**for**  $i = 1$  **to**  $iter$  **do**  
   Train the sampling network with resolution selected according to  $prob_1, \dots, prob_m$ ;  
**if**  $i \% (\frac{n}{2}) == 0$  **then**  
   **for**  $j = 1$  **to**  $m$  **do**  
      $errlist_j.insert(\mathcal{L}_{task}(X_i))$   
   **end for**  
**end if**  
**if**  $i \% (n + 1) == 0$  **then**  
   **for**  $j = 1$  **to**  $m$  **do**  
      $error_{nerr_j} = \frac{\max(errlist_j) - \min(errlist_j)}{\max(errlist_j)}$   
   **end for**  
   **for**  $j = 1$  **to**  $m$  **do**  
      $prob_j = \frac{\exp^{-nerr_j}}{\sum_1^m \exp^{-nerr_j}}$   
      $errlist_j \rightarrow []$   
   **end for**  
**end if**  
**end for**

---

where  $T_P$  and  $T_Q$  are reconstructed point clouds by task networks from  $P$  and  $Q$ , respectively.  $m$  is a margin to adjust the distillation degree. In details, the data distillation for reconstruction means to pay more attention to sampled results with relatively poor reconstruction performances against original point clouds. if  $m$  is bigger, more sampled results will be constrained. we set  $m = 0.001$ .

For point cloud classification, the task-oriented loss is cross-entropy defined as

$$\mathcal{L}_{CE}(z_S, q_S) = - \sum q_S \cdot \log(\text{softmax}(z_S)), \quad (4)$$

where  $q_S$  and  $z_S$  are the label and predicted result of point cloud  $S$ , respectively. Let  $P$  and  $Q$  be the original point clouds and sampled point clouds, following [12], the data distillation constraint for classification can be defined as

$$\mathcal{L}_{DD}(T_P, T_Q) = - \sum \text{softmax}(\frac{T_Q}{T}) \log(\text{softmax}(\frac{T_P}{T})), \quad (5)$$

where  $T_P, T_Q$  is the predicted outputs of target networks from original point clouds  $P$  and sampled results  $Q$ , respectively. The data distillation constraint for point cloud classification works by narrowing the distance between predicted distributions from point clouds before and after sampling.  $T$  is the temperature parameter to adjust the distribution of distillation constraint. we set  $T = 1.0$  in this work.

### 3.4 Simplification and Projection

**Simplification and projection.** In this work, we follow [16] to project generated sampled points to original point clouds. The projection constraint is defined as

$$\mathcal{L}_{proj} = t^2, \quad (6)$$

where  $t$  is a trainable parameter in the projection process. The simplification constraint [8, 16] is used to encourage the sampled points to be near from the original point clouds. Distances from points sets  $S_1$  to  $S_2$  can be defined as

$$\mathcal{L}_a(X, Y) = \frac{1}{|X|} \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2, \quad (7)$$

$$\mathcal{L}_m(X, Y) = \max_{x \in X} \min_{y \in Y} \|x - y\|_2^2. \quad (8)$$

Let  $Q, P$  be the sampled points and original point clouds, respectively. The simplification constraint can be defined as

$$\mathcal{L}_{sim}(Q, P) = \mathcal{L}_a(Q, P) + \beta \mathcal{L}_m(Q, P) + (\gamma + \delta |Q| \mathcal{L}_a(P, Q)) \quad (9)$$

Here, we follow the same setting for  $\beta, \gamma$  and  $\delta$  as [16].

**Move Constraint.** Our network acquires sampled points by driving initial seeds to new positions. The Move constraint can be presented as

$$\mathcal{L}_{mc} = \frac{1}{N} \sum \|Q_m\|_2, \quad (10)$$

where  $N$  and  $Q_m$  denote the sampling resolution and predicted displacement field as claimed in Sec. 3.1.

### 3.5 Loss function

Given all just defined components, the final training loss can be defined as

























$$\mathcal{L}_{final} = \mathcal{L}_{task} + \lambda_1 * \mathcal{L}_{DD} + \lambda_2 * \mathcal{L}_{sim} + \lambda_3 * \mathcal{L}_{proj} + \lambda_4 * \mathcal{L}_{mc}. \quad (11)$$

In this work, we set  $\lambda_1 \sim \lambda_4$  as 0.5, 1.0,  $10^{-5}$  and  $10^{-3}$ .

## 4 Experiments

### 4.1 Dataset and Implementation Details

In this work, we evaluate the sampling performances based on point cloud reconstruction and recognition. Three datasets: ShapeNet [4], ModelNet10 (MN10) and ModelNet40 (MN40) [30] are adopted for the training and evaluation processes. ShapeNet contains 12288 models in the train split and 2874 models in the test split from 16 categories following [31]. MN10 and MN40 are subsets of ModelNet, which contain 10

	GT-Rec/GT	Random	FPS	S-Net	SamNet	Ours
Reconstructed Results						
						
Sampled Points						
						

**Fig. 3.** Qualitative Comparisons between different sampling strategies under 32 sampled points. Sampled points are marked in blue. We can see that our method performs better on the circled thin and small regions by driving more sampled points to these areas.

categories and 40 categories of CAD models, respectively. All point clouds are composed of 2048 uniformly sampled points from mesh models.

For point clouds reconstruction, two commonly used baseline models AE [1] and FoldingNet [31] are introduced as the task networks. Task networks and sampling networks are both trained on the train split of ShapeNet. As for point cloud recognition, we train and evaluate the network performances on MN10 and MN40 [30] based on the task network PointNet [23] following [8, 16]. To make a fair comparison, S-Net [8] and SamNet [16] are trained with the progressive implements [8] to train a single model for all resolutions.

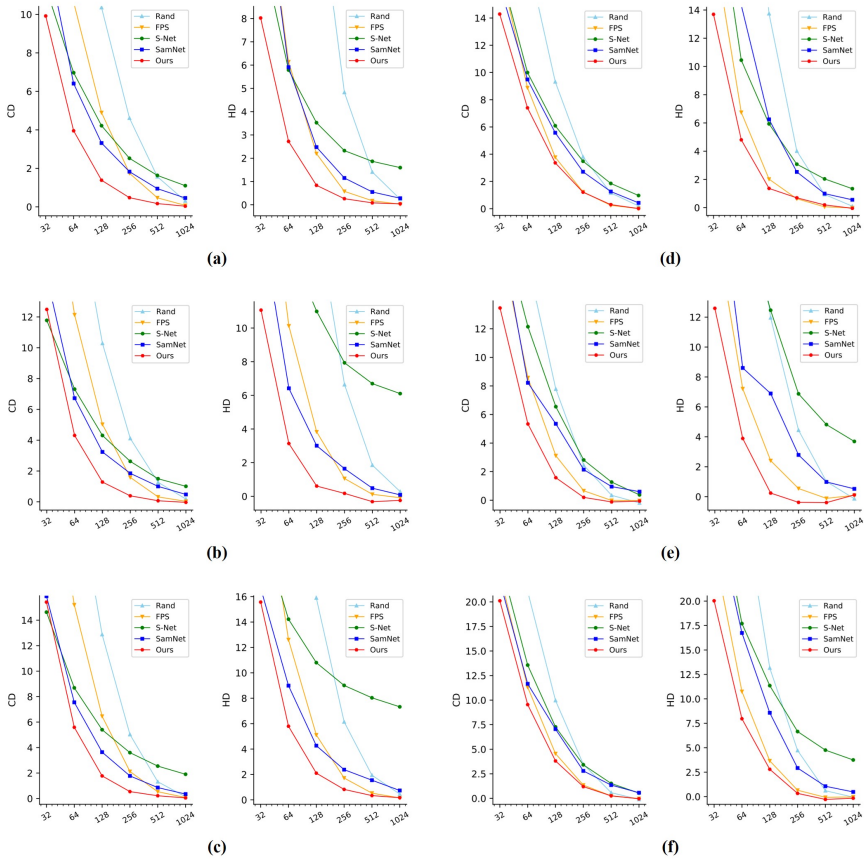
## 4.2 Experiments on Reconstruction Baselines

In this section, we evaluate the sampling performances on commonly-used reconstruction networks AE [1] and FoldingNet [31] pre-trained on train split of ShapeNet, while measuring reconstruction errors on the test split of ShapeNet, MN10 and MN40. The quantitative results are presented in Fig. 4. We evaluate sampling performances by increments of reconstruction errors compared to original point clouds, while the reconstruction errors are measured with Chamfer Distance (CD) [9] and Hausdorff Distance (HD) defined as

$$HD(S_1, S_2) = \frac{1}{2} (\max_{x \in S_1} \min_{y \in S_2} \|x - y\|_2 + \max_{x \in S_2} \min_{y \in S_1} \|x - y\|_2), \quad (12)$$

where  $S_1$  and  $S_2$  are two point clouds to be compared. CD and HD focus on the average and worst performances, respectively. We can see that our method can achieve lower reconstruction errors than existing sampling strategies on most adopted networks and datasets under different resolutions, which confirms that it is quite effective. To intuitively compare the performances of different sampling strategies, we also present a qualitative comparison in Fig. 3.



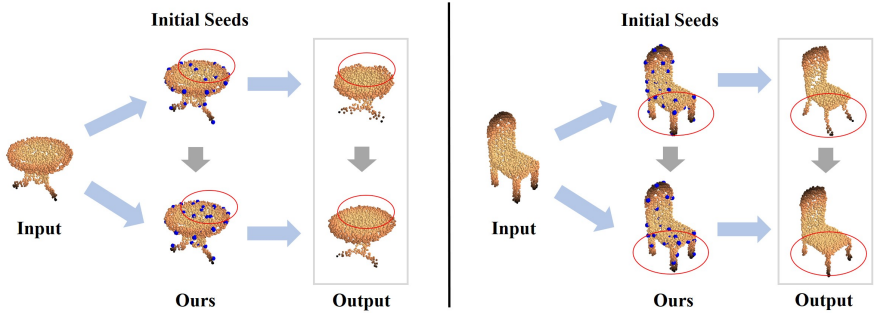


**Fig. 4.** Comparisons between sampling strategies. (a), (b) and (c) denote results on ShapeNet, MN10 and MN40 based on AE [1], while (d), (e) and (f) are results on ShapeNet, MN10 and MN40 based on FoldingNet [31]. All metrics are multiplied with  $10^3$ .

Reconstruction results from original point clouds and sampled points are presented in top 2 lines, while original point clouds and sampled results are presented in 2 lines below. Sampled points are marked in blue. We can see that existing methods pay little attention to some thin and small structures in original models as circled in Fig. 3, while our method can drive more sampled points to the error-prone regions and improve reconstruction performances.

### 4.3 Experiments on Recognition Baselines

For point clouds recognition, we train the learning-based sampling algorithms on MN10 and MN40 based on PointNet [23] following [8, 16]. The performances under different sampling resolutions are compared in Fig. 6. We can see that our method can achieve higher classification accuracy than other sampling algorithms on both MN10



**Fig. 5.** Visualization of the way PCDNet works. Initial seeds from non-learning-based sampling strategies and final sampled points from our PCDNet are marked in blue. We feed them to a same AE [1] to observe their reconstructed outputs.

and MN40. An interesting condition is that our method can even exceed the classification accuracy of original point clouds on MN40, while all learning-based sampling networks get higher accuracy than original point clouds on MN10 under the resolution of 512 points.

The recognition network actually works by extracting specific structural features from original point clouds. Some points may be redundant for features, which actually introduce noises for the recognition network. Sampling can be regarded as a filtering process to remove noises from original points clouds. There is a trade-off between the sampling resolution and noises. Sampled points are less affected by noises under a lower resolution, while the structural features are also limited by less points. As the resolution increases, impacts from original noises and structural features contained in sampled points are both increased.

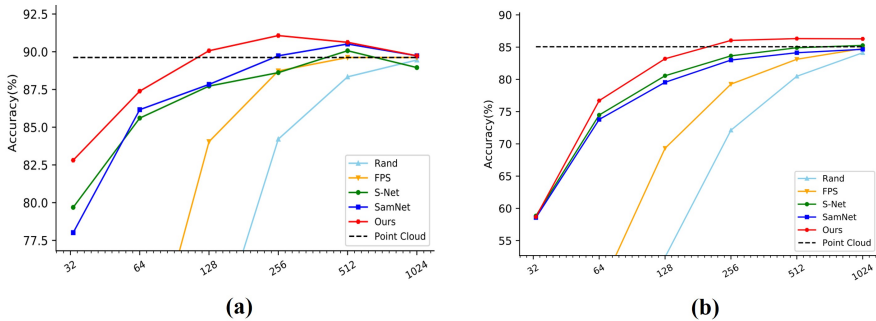
In this condition, sampled points under a certain resolution may overstep original point clouds on specific task network with an end-to-end optimization. Higher resolution will instead reduce the performances, as shown in Fig. 6-(a). Note that our method in Fig. 6-(b) has 86.32% accuracy at 512 points and 86.28% accuracy at 1024 points, which is also consistent with our analysis.

#### 4.4 How does PCDNet work?

PCDNet actually works by driving original sampled points to more appropriate positions. We visualize the points before and after the movements to see how PCDNet works. The results are presented in Fig. 5. We can see that initial seeds from FPS may create defective regions as shown by the circled areas. PCDNet can learn to drive more sampled points to the failed areas, which can introduce more structural details from these areas.

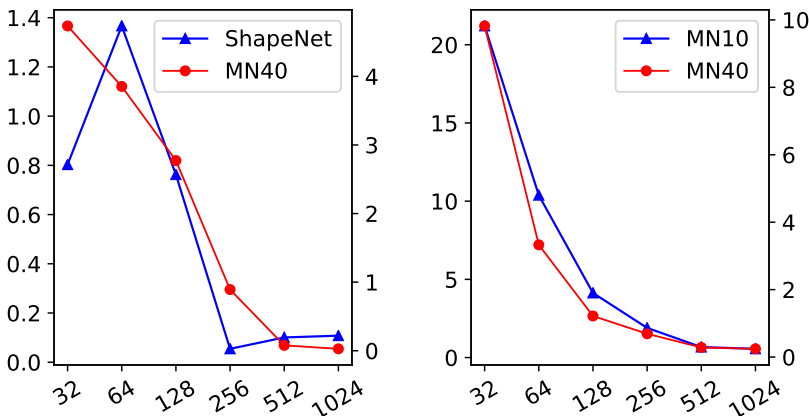
#### 4.5 Analysis of the Data Distillation

To observe the behaviors of data distillation, we present its improvements on reconstruction and recognition tasks under different resolutions. The results are presented



**Fig. 6.** Comparisons on recognition. (a) and (b) denote results on MN10 and MN40, respectively.

in Fig. 7. We can see that the data distillation brings more improvements on low resolutions. The reason is that the data distillation is based on differences between task-oriented performances of sampled points and original point clouds. It can provide stronger supervision when the performances of sampled points are quite different with original point clouds. We can see that data distillation can help our sampling network achieve 20% higher classification accuracy on MN10 and 10% improvements on MN40 under 32 sampled points, which confirms that it is quite effective.

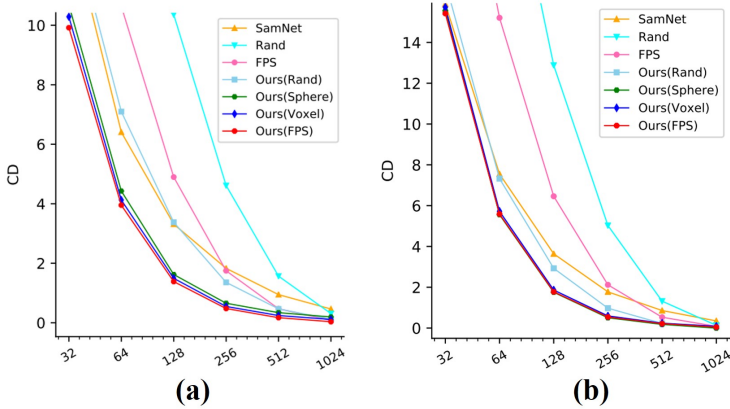


**Fig. 7.** Improvements from data distillation under different resolutions, i.e. the 0.0 denotes results of PCDNet trained without data distillation. left and right vertical axes denote metrics measured on datasets marked in blue and red, respectively.

#### 4.6 Sensitivity to the Non-learning Sampling

In this work, we use FPS to generate initial sampled points. To explore the sensitivity of our method to the non-learning sampling strategy, we conduct a group of comparisons

on AE [1] based on a few commonly-used sampling strategies including random sampling, FPS, Voxel Downsampling, and Sphere filtering. The SOTA method SamNet [16] is also introduced to make a comparison. The results are presented in Fig. 8.



**Fig. 8.** Comparisons between different initial sampling strategies. (a) and (b) denote comparisons on ShapeNet and ModelNet40, respectively.

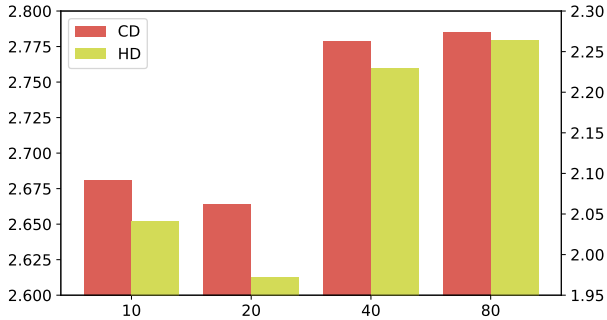
We can see that the performances based on FPS, Voxel Downsampling, and Sphere filtering are quite close, which confirms that our framework is robust when the initial sampled points have relatively uniform spatial distribution. The performances based on random sampling has obvious decline compared to other strategies due to its randomness. Intuitively speaking, the randomly sampled results may be quite imbalanced to cover the whole shapes, which makes it hard to drive them to well-performed positions. But even results based on random sampling has slightly better performances on ModelNet40 than SamNet, which confirms our framework is effective.

#### 4.7 Comparisons of Sampling Efficiency

In this section, we compare the inference time and memory cost between different learning-based sampling strategies on AE [1]. The inference time and memory cost are measured by the average time and memory cost under different resolutions between 32 ~ 1024. The results are presented in Table 1. We can see that our method has lower

Methods	S-Net	SamNet	Ours
Time(ms)	9	17	<b>7</b>
Memory(MB)	842	1156	<b>797</b>
Parameter(M)	1.77	1.77	<b>0.28</b>

**Table 1.** Model efficiency comparison. Time and memory are evaluated on a Nvidia 2080ti GPU with a 2.9Ghz i5-9400 CPU.



**Fig. 9.** Ablation study for the resolution updating interval. left and right vertical axes denote CD and HD multiplied by  $10^3$ .

time cost, less memory cost and smaller parameter than existing learning-based sampling strategies S-Net [8] and SamNet [16].

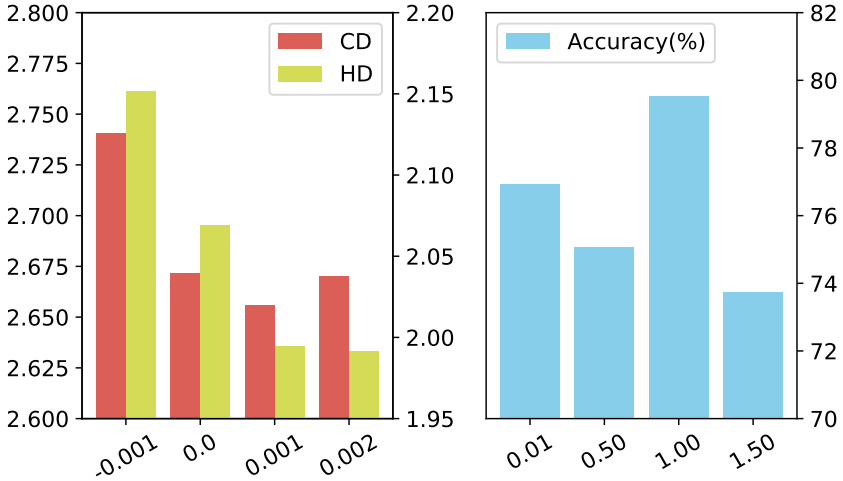
#### 4.8 Ablation Study

**Influence of Components in PCDNet.** To clarify the influence of each component in PCDNet, we conduct ablation experiments in this section. Dyna, Reso, Move and Dist are Dynamic resolution selection, Resolution-aware pooling, Move constraint and Data distillation, respectively. Base denotes the sampling network without Resolution-aware pooling, which is trained with task-oriented loss, simplification constraint and projection operation following [16]. CD\* denotes CD metric measured under 32 sampled points, while CD and HD are average metrics under different resolutions between  $32 \sim 1024$ . We can see each module makes sense. Removing any module will reduce the performances.

BASE	RESO	DIST	DYNA	MOVE	CD	HD	CD*
✓					3.70	3.31	12.45
✓	✓				2.99	2.50	10.74
✓	✓	✓			2.89	2.12	10.36
✓	✓	✓	✓		2.74	2.01	10.05
✓	✓	✓	✓	✓	<b>2.67</b>	<b>1.97</b>	<b>9.91</b>

**Table 2.** Ablation study for components. All metrics are average values under different resolutions, which are multiplied with  $10^{-3}$ .

**Influence of resolution updating interval.** The number of dynamic resolution updating interval  $n$  mentioned in Sec. 3.2 has influence on the final performance. A smaller interval will lead the network to change resolutions frequently, which may introduce more randomness. Though a big interval can reduce the randomness when evaluating the training errors, the selection frequency may be not high enough to train all resolutions well. In this section, we conduct a series of experiments to observe the influences



**Fig. 10.** Ablation study for the distillation hyper-parameters. left and right vertical axes measure CD and HD multiplied by  $10^3$ .

of different resolution updating intervals. The results are presented in Fig. 9. We can see that 20 is a good choice.

**Influence of distillation hyper-parameters.** Data distillation is an interesting component proposed to improve the network performances by introducing extra supervision from original network outputs. Hyper-parameters such as  $m$  for reconstruction networks and  $T$  for recognition networks defined in Sec. 3.3 are used to adjust the influence of original outputs. Smaller  $T$ , e.g., will increase the impacts of both existing knowledge and misclassified noises for recognition networks. To figure out the influences of these hyper-parameters, we present the ablation experiments performances in Fig. 10. We can see that the reconstruction network achieves the lowest error when  $m = 0.001$ , while the recognition network gets highest accuracy at  $T = 1.0$ .

## 5 Conclusion

In this work, we propose a new resolution-free point cloud sampling network to deal with different resolutions with a same architecture. By driving initial seeds sampled with non-learning-based sampling strategies such as FPS, we can directly sample original point clouds to any resolution by adjusting the resolution of initial seeds. Besides, we propose data distillation to assist the optimization by considering differences between task network outputs from sampled points and original point clouds based on knowledge distillation constraints. Experiments on reconstruction and recognition demonstrate that our method achieves better task-oriented performances with lower time and memory cost than existing learning-based sampling networks.

## References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International conference on machine learning. pp. 40–49. PMLR (2018)
2. Ahn, S., Hu, S.X., Damianou, A., Lawrence, N.D., Dai, Z.: Variational information distillation for knowledge transfer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9163–9171 (2019)
3. Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J.: Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics* **32**(6), 1309–1332 (2016)
4. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015)
5. Chen, G., Choi, W., Yu, X., Han, T., Chandraker, M.: Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems* **30** (2017)
6. Chen, H., Wang, Y., Xu, C., Yang, Z., Liu, C., Shi, B., Xu, C., Xu, C., Tian, Q.: Data-free learning of student networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3514–3522 (2019)
7. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence* **40**(4), 834–848 (2017)
8. Dovrat, O., Lang, I., Avidan, S.: Learning to sample. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2760–2769 (2019)
9. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017)
10. Fang, G., Song, J., Shen, C., Wang, X., Chen, D., Song, M.: Data-free adversarial distillation. *arXiv preprint arXiv:1912.11006* (2019)
11. Heo, B., Lee, M., Yun, S., Choi, J.Y.: Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 3779–3787 (2019)
12. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015)
13. Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A.: Randlanet: Efficient semantic segmentation of large-scale point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11108–11117 (2020)
14. Huang, T., Zou, H., Cui, J., Yang, X., Wang, M., Zhao, X., Zhang, J., Yuan, Y., Xu, Y., Liu, Y.: Rfnet: Recurrent forward network for dense point cloud completion. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12508–12517 (2021)
15. Huang, Z., Wang, N.: Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv preprint arXiv:1707.01219* (2017)
16. Lang, I., Manor, A., Avidan, S.: Samplenet: Differentiable point cloud sampling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7578–7588 (2020)
17. Li, J., Chen, B.M., Lee, G.H.: So-net: Self-organizing network for point cloud analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9397–9406 (2018)

18. Li, R., Li, X., Fu, C.W., Cohen-Or, D., Heng, P.A.: Pu-gan: a point cloud upsampling adversarial network. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 7203–7212 (2019)
19. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems* **31**, 820–830 (2018)
20. Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8895–8904 (2019)
21. Lopes, R.G., Fenu, S., Starner, T.: Data-free knowledge distillation for deep neural networks. arXiv preprint arXiv:1710.07535 (2017)
22. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9277–9286 (2019)
23. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
24. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: *Advances in neural information processing systems*. pp. 5099–5108 (2017)
25. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550 (2014)
26. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.H., Kautz, J.: Splatnet: Sparse lattice networks for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2530–2539 (2018)
27. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: Kpconv: Flexible and deformable convolution for point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6411–6420 (2019)
28. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)* **38**(5), 1–12 (2019)
29. Wu, W., Qi, Z., Fuxin, L.: Pointconv: Deep convolutional networks on 3d point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9621–9630 (2019)
30. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1912–1920 (2015)
31. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 206–215 (2018)
32. Yin, K., Chen, Z., Huang, H., Cohen-Or, D., Zhang, H.: Logan: Unpaired shape transform in latent overcomplete space. *ACM Transactions on Graphics (TOG)* **38**(6), 1–13 (2019)
33. Yu, L., Li, X., Fu, C.W., Cohen-Or, D., Heng, P.A.: Pu-net: Point cloud upsampling network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2790–2799 (2018)
34. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. arXiv preprint arXiv:1612.03928 (2016)