Pattern Recognition Letters 164 (2022) 216–223

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Fast Point Cloud Sampling Network

Tianxin Huang[a], Jun Chen[a], Jiangning Zhang[a], Yong Liu[a,*], Jie Liang[a,b]

[a] Laboratory of Advanced Perception on Robotics and Intelligent Learning, College of Control Science and Engineering, Zhejiang University, Hangzhou, China
[b] Beijing Institute of Mechanical and Electrical Engineering, Beijing, China

ARTICLE INFO

Article history:
Received 14 January 2022
Revised 26 September 2022
Accepted 7 November 2022
Available online 16 November 2022

Edited by Jiwen Lu

Keywords:
3D Point Cloud
Neural Network
Sampling

ABSTRACT

The increasing number of points in 3D point clouds has brought great challenges for subsequent algorithm efficiencies. Down-sampling algorithms are adopted to simplify the data and accelerate the computation. Except the well-known random sampling and farthest distance sampling, some recent works have tried to learn a sampling pattern according to the downstream task, which helps generate sampled points by fully-connected networks with fixed output point numbers. In this condition, a progress-net structure covering all resolutions sampling networks or multiple separate sampling networks for different resolutions are required, which is inconvenient. In this work, we propose a novel learning-based point cloud sampling framework, named Fast point cloud sampling network (FPN), which drives initial randomly sampled points to better positions instead of generating coordinates. FPN can be used to sample points clouds to any resolution once trained by changing the number of initial randomly sampled points. Results on point cloud reconstruction and recognition confirm that FPN can reach state-of-the-art performances with much higher sampling efficiency than most existing sampling methods.

1. Introduction

With the rapid development of 3D tasks, point cloud has attracted more and more attentions in computer vision and robots. However, larger and larger point clouds may limit the efficiency of related algorithms and ask for more advanced equipment. Sampling the original point clouds to lower resolutions is an alternative solution to reduce the computational cost. Existing works [1–3] often use random sampling and the farthest point sampling (FPS) to down-sample the point clouds. However, the direct sampling methods do not consider down-stream tasks, which limits their further improvements on task performances. In this condition, some researchers [4,5] introduce task-oriented optimizations to guide the sampling process. S-Net [5] generates sampled points directly by fully-connected networks. Sampled points would be fed into a pre-trained task network to provide a task-oriented loss. In this way, the performances of sampled points on specific task network will be improved. To provide supports for different resolutions, [5] also proposes a progress-net structure to generate sampled points with the same number of input points and choose points from all generated points according to the resolution. Points of multiple sampling resolutions are then fed into the task network to get constraints for all resolutions. However, the progress-net structure actually trains all resolution networks together, which increases the computational cost and may be inconvenient. To prevent sampled points from deviating original shapes, [5] projects generated points to their nearest neighbors in original models. The task performances would decrease after projection because the projection is not considered during training. SamNet [4] gets over this problem by differentiable projection of the generated points to the original point sets. But it still needs multiple repeated training processes or a large progress-net structure for sampling under different resolutions.

In this work, we propose a simple but effective solution for the sampling under relatively high resolutions where the efficiency should be considered, named the Fast Point cloud sampling Network (FPN). The differences between our work and former learning-based works are presented in Fig. 1.

We sample points by driving initial randomly sampled points from non-learning based sampling strategies to better positions instead of generating points directly, which is more concise and efficient. The discrepancy between progress-net and our method is presented in Fig. 1-(b) and (c). The progress-net implements for S-Net [5] and SamNet [4] achieve multi-resolution sampling by generating points with highest resolutions and choose the first M ones as M sampled points, while our method can naturally support sampling under different resolutions by changing the resolution of initial points. We can see that progress-net always generates useless sampled points, while our method can avoid the redundancy

* Corresponding author.
E-mail addresses: 21725129@zju.edu.cn (T. Huang), junc@zju.edu.cn (J. Chen), 186368@zju.edu.cn (J. Zhang), yongliu@iipc.zju.edu.cn (Y. Liu), liangjie@iipc.zju.edu.cn (J. Liang).

https://doi.org/10.1016/j.patrec.2022.11.006
0167-8655/

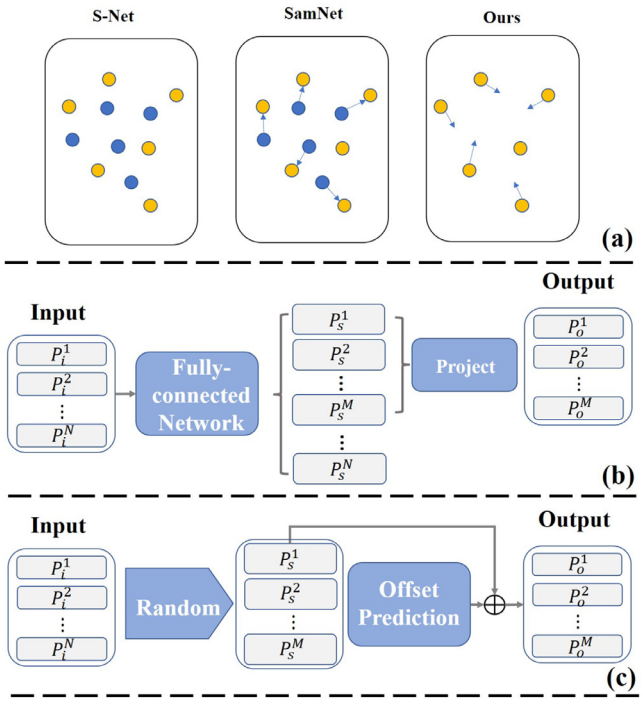

Fig. 1. (a) shows the differences between learning-based sampling strategies, while (b) and (c) present the discrepancy between progress-net and our method in multi-resolution sampling. P_i, P_o, and P_s are input, output sampled points and intermediate coordinates, respectively. We assume that we sample M points from N input points. Blue points and yellow points denote generated and original points, respectively. S-Net generates sampled points directly, while SamNet projects generated points to original models by differentiable projection during training. Our work adopts a different operation to drive existing points instead of generation.

and further improve the sampling efficiency. Besides, we introduce a Hybrid Training Strategy (HTS) to help FPN adapt to multiple sampling resolutions by randomly introducing different resolution sampled points during training. Sampled points are guaranteed to meet the shape of original point clouds by constraining the driving distances without direct projection to avoid the performances reduction. Progress-net optimizes losses of multiple resolutions in a iteration to constrain all N sampled points, which has large computational cost. By applying HTS in FPN, we can train the network resolution-by-resolution with relatively low computation cost. Our contributions can be summarized as:

- We propose a novel learning-based point cloud sampling framework named fast sampling network (FPN) by driving existing randomly sampled points to better positions;
- We introduce a hybrid training strategy to help FPN adapt to different sampling resolutions by randomly introducing selecting the resolution of initial points during training;
- The results on point cloud reconstruction and recognition confirm that FPN outperforms the existing point clouds sampling strategies with high sampling efficiency.

2. Related Works

2.1. Point Cloud Learning

Early works [6–8] usually apply 3D CNNs based on voxel representation of 3D point clouds. However, 3D volumes can not be directly acquired. Converting point clouds to 3D voxels is expensive, which also leads to quantization errors caused by losing some details of the original data. So Qi et al first introduced a point-based point cloud learning network named PointNet [9]. It processes point clouds with multilayer perceptrons (MLPs) and aggregates features with symmetric functions. PointNet++ [1] captures local features by recurrently applying PointNet in local regions acquired by ball query around sampled points. Lots of works have been proposed based on PointNet and PointNet++ such as the point cloud analysis [10–13], reconstruction [14–16] or up-sampling [17,18]. DGCNN [19] builds dynamic graphs by selecting neighbors with distances between point features, which gets over the distance limitation of PointNet++. Except methods extracting features directly based on MLPs, some methods such as PointConv [20] and KPconv [21] use MLPs to assign weights for points and design corresponding convolution methods.

2.2. Sampling Strategies

Random sampling and farthest point sampling (FPS) [1] 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, some learning-based works [4,5] have also been released to enhance the performance of sampled points for specified tasks such as recognition, reconstruction and registration. They are optimized with pre-trained task networks with task-oriented losses. However, the learning-based methods are still quite limited by the network designation. [5] adopts fully-connected networks to help generate sampled points, while [4] differentiable projects the generated points to the original point clouds to get more practical points distribution. Both of them need to train a separate network for a resolution. Though [5] presents progress-net to get sampling networks for all resolutions in a single training process, it actually trains a big generation network contains all possible resolutions, which generates extra unused points and may be quite inconvenient.

3. Methodology

3.1. Basic pipeline

The basic pipeline of FPN is presented in Fig. 2. We aggregate global features from the input points with a set of multilevel perceptions (MLPs) and Max Pooling following PointNet [9]. The global features would contain information of original models. Then we sample some initial points with random sampling, which would be concatenated with global features and combined into merged features including information from both initial sampled points and original models. Finally, the initial sampled points will be concatenated again with merged features and fed into MLPs to predict an offset for each initial point. By moving initial points with predicted offsets, FPN can get sampled points with any resolution with a simple and efficient network structure.

If we define the input as X, output as Y, random sampling operation as $s(\cdot)$, concatenation operation as $C(\cdot)$, and the PointNet structure as $f(\cdot)$, we can present FPN as

$$\begin{cases} s_0 = s(X, r), \\ Y = s_0 + MLPs(C(f(C(f(X), s_0)), s_0)), \end{cases} \tag{1}$$

where s_0 is the initial randomly sampled points with the resolution of r. We can see that the final sampling resolution is only decided by the resolution of initial random sampling, which means that a universal sampling network can be learned for all resolutions. By simply selecting different numbers of initial points, the sampling resolution of FPN can be easily adjusted without changing the network organization.

3.2. Hybrid Training Strategy

The FPN structure supports sampling different resolution points by predicting offsets for different number of initial points. We introduce hybrid training strategy (HTS) to help FPN adapt to different resolutions, while most existing learning-based works [4,5] require multiple repeated training processes or redundant networks to sample point clouds to different resolutions as we mentioned in Section 1. The achievement of HTS is presented as Algorithm 1. The resolution of initial points is randomly selected between resolutions from N_1 to N_2 during each training iteration, which is adopted to improve the robustness of FPN on multiple resolutions.

3.3. Loss function

The loss fuction of FPN consists of two parts: the task-oriented loss and range constraint. The task-oriented loss is adopted to guide the sampling process, while range constraint is introduced to prevent the sampled points from going too far from the initial points. The sampled results may deviate the original contour if sampled points are allowed to go too far. The range constraint can be presented as

$$\mathcal{L}_{rc} = \frac{1}{N}\sum \|S_o - S_i\|_2, \tag{2}$$

where N is the sampling resolution. S_i and S_o denote initial sampled points and final sampled points, respectively. The final training loss would be

$$\mathcal{L}_{loss} = L_{task} + \lambda L_{rc}, \tag{3}$$

while L_{task} denotes the task-oriented loss. For reconstruction-related tasks, it may be Chamfer Distance or Earth Mover Distance [22] defined as

$$\begin{aligned}\mathcal{L}_{task} &= \mathcal{L}_{CD}(S_1, S_2)\\ &= \frac{1}{2}\left(\frac{1}{|S_1|}\sum_{x\in S_1}\min_{y\in S_2}\|x-y\|_2 + \frac{1}{|S_2|}\sum_{x\in S_2}\min_{y\in S_1}\|x-y\|_2\right),\end{aligned} \tag{4}$$

or

$$\mathcal{L}_{task} = \mathcal{L}_{EMD}(S_1, S_2) = \min_{\phi:S_1\to S_2}\frac{1}{|S_1|}\sum_{x\in S_1}\|x-\phi(x)\|_2, \tag{5}$$

where S_1 and S_2 are input and output. ϕ is a bijection from S_1 to S_2. For classification-related tasks, the task-oriented loss may be cross entropy defined as

$$\mathcal{L}_{task} = \mathcal{L}_{CE}(p, q) = -\sum p(x)log(q(x)), \tag{6}$$

where x is the data, p, q are the label and predicted distributions.

4. Experiments

4.1. Dataset and implementation details

In this work, we train FPN on ShapeNet part dataset [23]. ShapeNet part dataset consists of 12288/1870/2874 models in the train/val/test splits from 16 categories following [24,25]. ModelNet10 and ModelNet40 are subsets of ModelNet [26], which contain 10 and 40 categories models, respectively. ShapeNet, ModelNet10 and ModelNet40 are all datasets composed of manually reconstructed CAD models. All point clouds consist of 2048 points uniformly sampled on mesh models.

We conduct comparisons on point cloud recognition and reconstruction tasks for object models to evaluate the performance of FPN. Randomly sampling, farthest point sampling (FPS), S-Net [5] and SamNet [4] are adopted to compare. To provide a fair comparison, S-Net [5] and SamNet [4] are trained with the progress-net method. To train FPN, we consider the sampling with relatively high resolutions between 64 $\sim$ 1024 following settings of [1,3,15,17]. The network is trained with Adam optimizer under a learning rate 0.0001.

Table 1
The number of neurons in networks. f_1, f_2, f_3 are modules in Fig. 2.

	f_1	f_2	f_3
MLPs	(128,256,256)	(128,256,256)	(128,128,3)

Table 2
The comparison on optimal clustering.

Center	Iterations	1	10	100
16	FPS	2.43	2.00	1.98
	Ours	**2.16**	**1.98**	**1.96**
32	FPS	1.20	1.02	1.00
	Ours	**1.11**	**1.00**	**1.00**

All experiments and evaluations are conducted on a NVIDIA 2080ti GPU with a 2.9GHZ i5-9400 CPU based on Tensorflow. The hyper-parameter λ is tuned on the validation split of ShapeNet. Detailed network structures are shown in Table 1.

4.2. Comparisons on reconstruction

In this section, we evaluate the sampling performance based on the reconstruction network [25] pre-trained on ShapeNet and measure reconstruction errors on ModelNet10 and ModelNet40. The qualitative and quantitative results are presented in Fig. 3 and Fig. 4. Fig. 4-(a) and (b) compare performances of reconstruction networks trained and evaluated based on CD, while Fig. 4-(c) and (d) present results trained and evaluated based on EMD. We can see that our method achieves the lowest reconstruction errors under different resolutions. To provide a more intuitive comparisons, we also choose a few models from the test data to present a qualitative comparison in Fig. 3 by illustrating the reconstruction results of different sampling strategies under 128 sampled points. We can see that existing sampling methods may create unexpected defects on reconstructed results at some continuous regions such as sofa legs and chair backs as circled, while our method can get over the flaws by driving more points to the edges of these circled regions.

4.3. Comparisons on real scans

As shapeNet, ModelNet10 and ModelNet40 are all synthetic data, we also conduct experiments on objects in real scans from ScannNet in [27] to further explore the effectiveness of FPN. The results are presented in Fig. 6. We can see that FPN shows clearer improvements over existing learning-based methods S-Net and SamNet, which confirms its great robustness.

4.4. Comparisons on recognition

For point cloud recognition [9], we evaluate classification accuracy on commonly-used ModelNet10 and ModelNet40 under different sampling resolutions. PointNet [9] networks pre-trained on corresponding datasets are adopted as the task networks. The results are presented in Fig. 5. We can see that our method achieves the highest accuracy on almost all resolutions, which proves the effectiveness of FPN.

4.5. Discussion about clustering

Except down-stream tasks such as reconstruction or recognition, down-sampled points can also be adopted as the initial clustering centers. The farthest point sampling (FPS) is an approximation of centers in k-center clustering problem [28], while K-means++ is actually a probabilistic implementation of FPS by giving

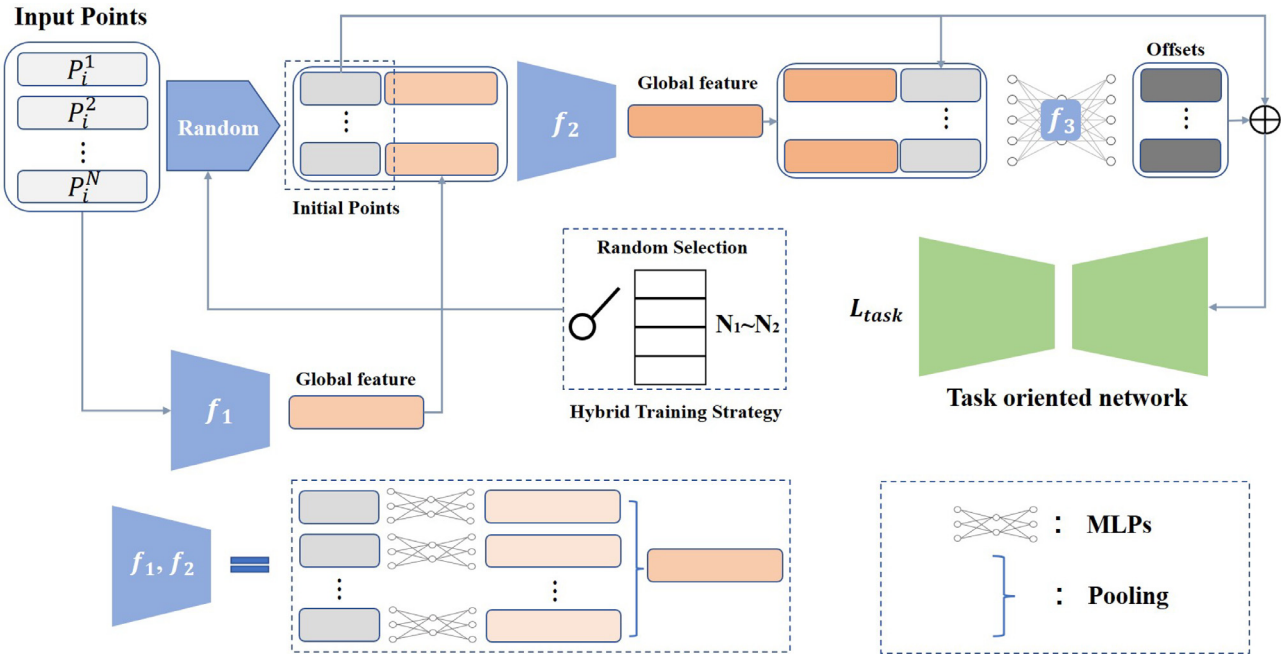

Fig. 2. The whole pipeline of FPN. The + denotes element-wised addition. f_1 and f_2 aggregate features by MultiLayer Perceptrons(MLPs) and pooling, while f_3 is a group of MLPs to predict offsets from coordinates and features. The task network is corresponding to the specific task, such as point cloud recognition and reconstruction. L_{task} is the loss constrained the task network.

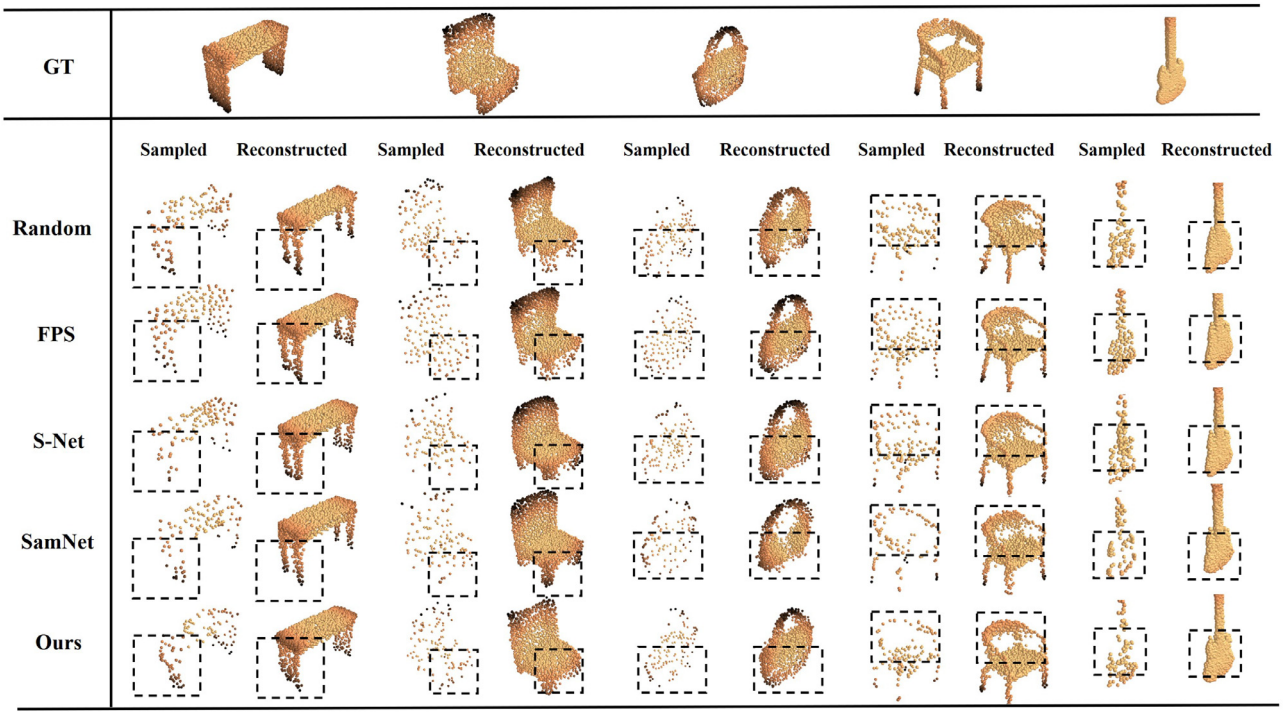

Fig. 3. Qualitative comparisons of different sampling strategies. The main differences are circled for clearer demonstration.

higher probabilities for farther points. In this section, we compare performances of optimal clustering between initial centers sampled with FPS and FPN initialized with FPS. The clustering performances are measured by average distances between all points and their clustering centers, while FPN is trained by optimizing the distances between all points and sampled points. The results are presented in Table 2. We can see that our network can still enhance FPS for a better initialization of clustering, which shows more obvious improvements when the iterations of K-means are smaller. As the iteration increases, the performances gradually become close because the influence of initialization becomes weaker.

4.6. Comparisons on sampling efficiency

The sampling efficiency is important for real-world applications. In this section, we compare the time efficiency of different

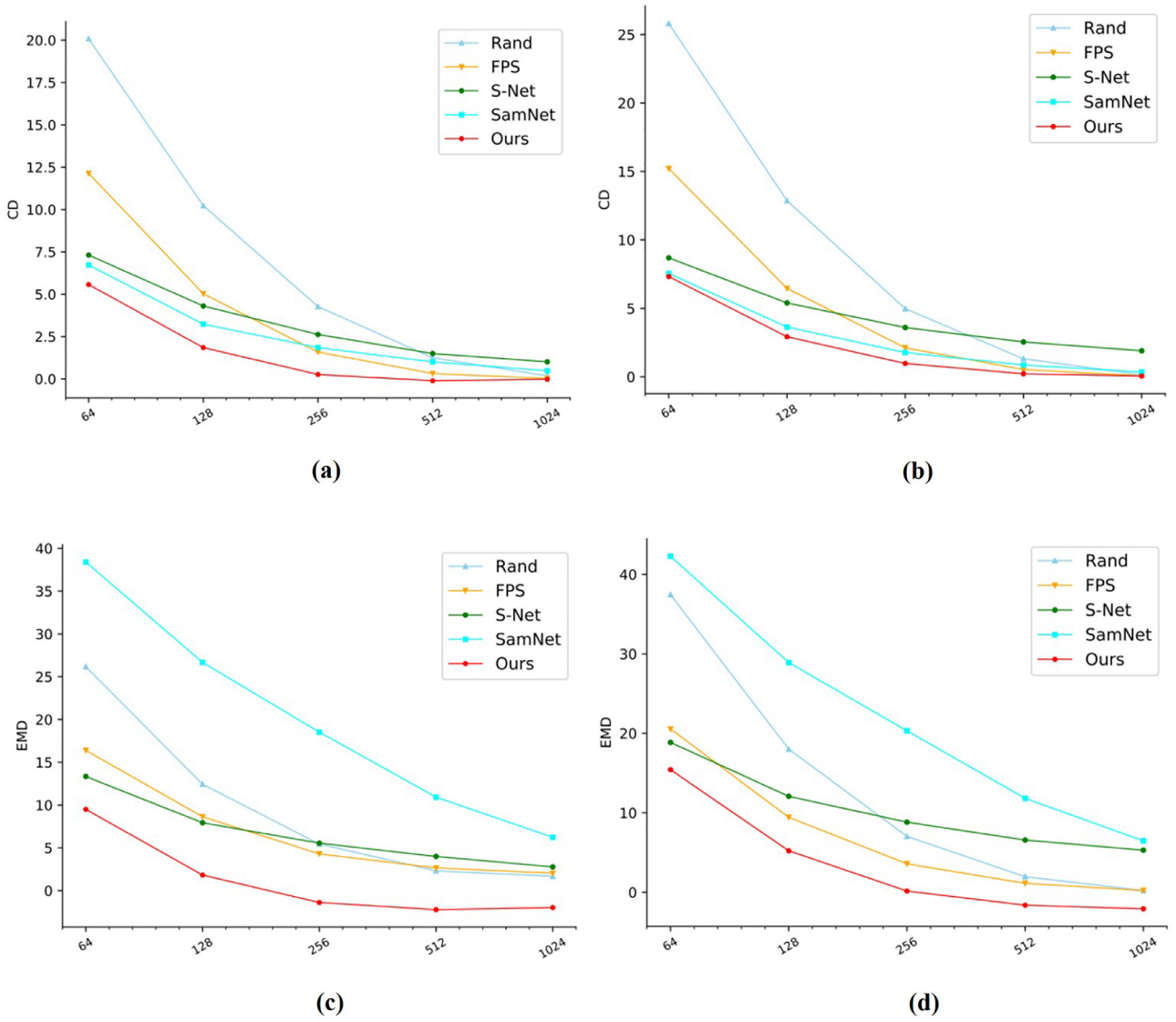

Fig. 4. Reconstruction comparisons between sampling strategies. (a) and (b) denote errors evaluated on ModelNet10 and ModelNet40 for CD-based reconstruction networks, while (c) and (d) show performances of EMD-based reconstruction networks on ModelNet10 and ModelNet40.

sampling strategies under different resolutions between 64∼1024. Though random sampling is a little faster than our method, it often gets the worst results as shown in Fig. 4 and Fig. 5. FPN outperforms commonly used sampling strategies such as FPS and S-Net on task performances, while it is only slower than the random sampling. From Fig. 8 we can see that FPN is much faster than S-Net [5] and FPS, which shows its high sampling efficiency.

4.7. How does FPN works?

FPN actually works by driving the initial points sampled by random sampling. It would be interesting to observe what does FPN do to the initial points. In this section, we present a visualization in Fig. 7. We visualize initial and final sampled points as well as their reconstruction results based on a pre-trained reconstruction network [25]. We can see that the initial random sampling may miss some shapes such as the table leg or the airplane tail due to the randomness, while FPN can help random sampling more robust to cover the whole shape. Then the reconstruction performance of FPN is improved on the missed part.

In short, FPN learns a pattern to change the distribution of initial points according to the specific task, which can get over weaknesses and uncertainty of randomly sampled initial points.

4.8. Ablation Study

Comparison between different sampling organizations. FPN gets sampled points by driving existing sampled points to better positions, while former learning-based S-Net and SamNet generate points with fully-connected networks. To compare the driving network and fully-connected network, we conduct a group of experiments between the driving and fully-connected networks. The results are presented in Fig. 9. We can see that the driving network outperforms the fully-connected network especially on higher resolutions. It may be difficult for fully-connected networks to generate relatively dense points, while our driving network is always robust for all resolutions.

The influence of range constraint. We conduct an ablation study for λ. Note that this is only conducted to observe the influence of range constraint weight λ on sampling performances instead of the tuning of λ, which is chosen according to the val set

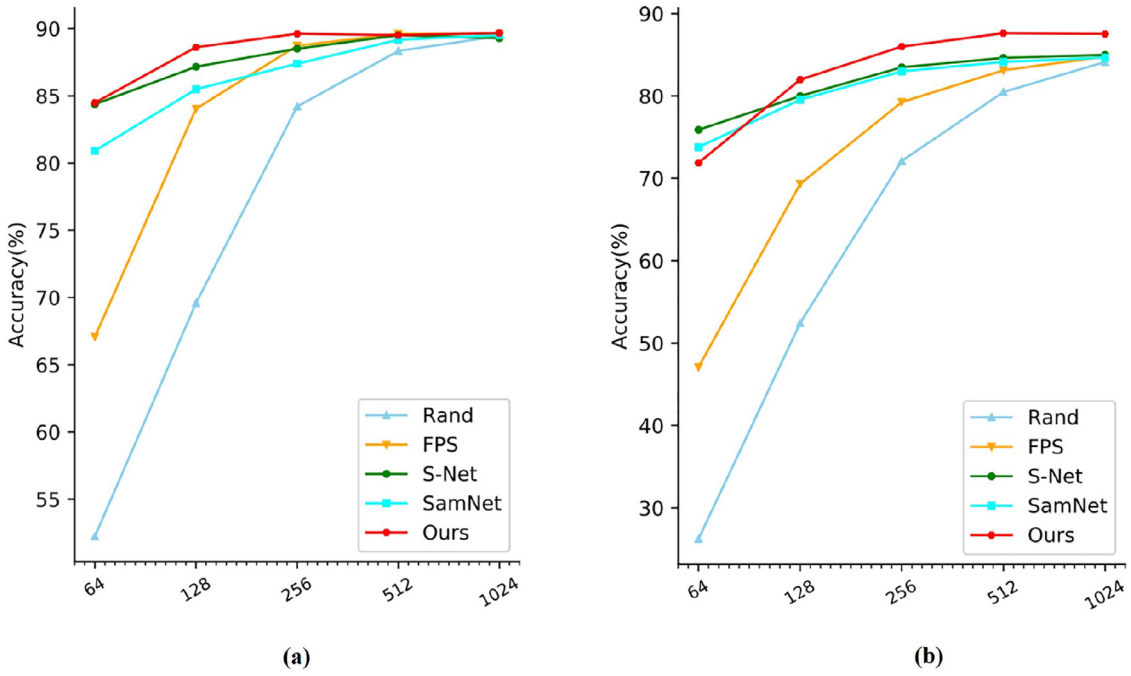

Fig. 5. Recognition comparisons between sampling strategies. (a) and (b) denote errors evaluated on ModelNet10 and ModelNet40, respectively.

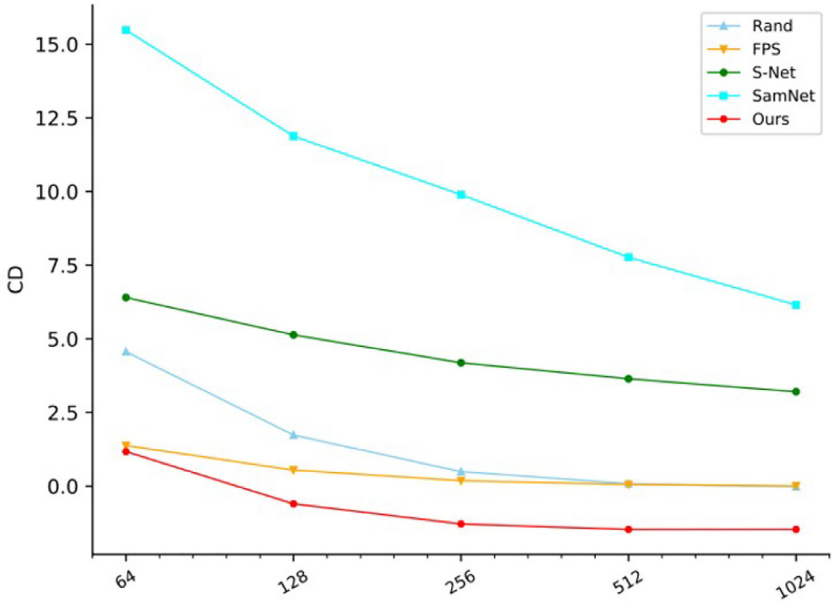

Fig. 6. Comparisons of reconstruction networks trained on real scans.

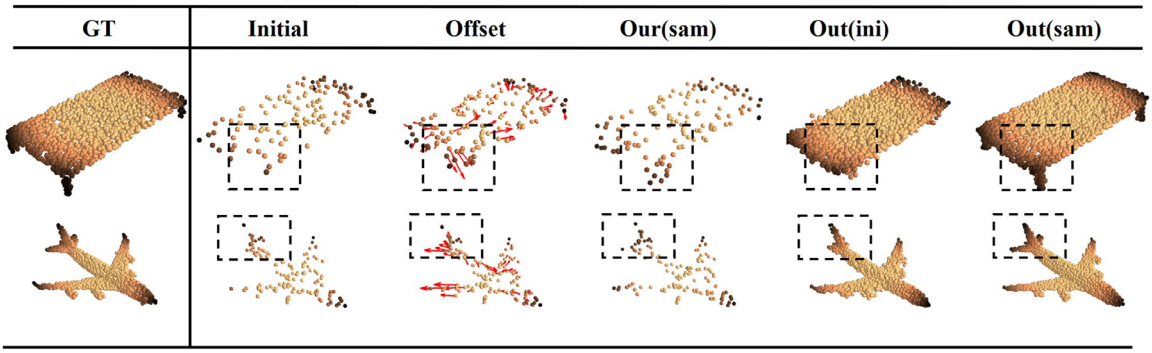

Fig. 7. Visualization of how our network takes effect. Initial randomly sampled points and our driven results are fed into same pre-trained reconstruction network to observe the performances. Initial and Ours(sam) denote initial and driven sampled points, while offset shows their ways to move. Out(ini) and Out(sam) show the reconstruction results with initial and our sampled points, respectively.

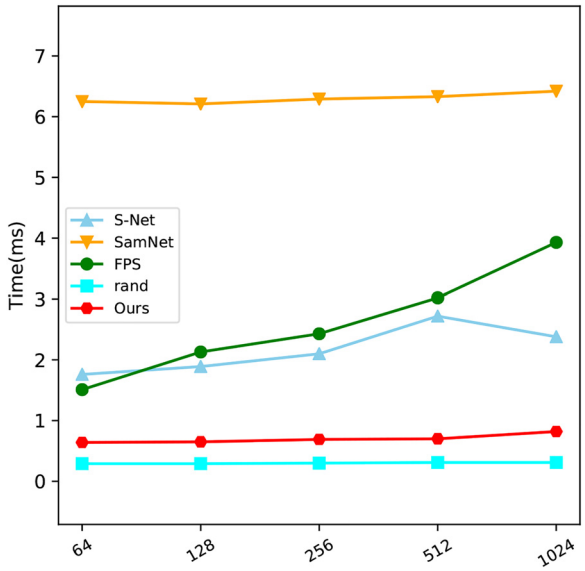

Fig. 8. Sampling efficiency comparisons.

introduced in Section 4.1. All experiments are conducted by repeatedly training FPN based on parameter-fixed AE [25] following S-Net and SamNet under different settings. The results are presented in Fig. 10. We can see that too big or small λ both have negative influence. Bigger weights limit the driving of initial sampled points too much to get over all weaknesses, while smaller weights may cause sampled points to go too far from original shapes and reduce the performances instead.

Algorithm 1 Training with Hybrid Training Strategy

Input: data X, the number of iterations *iter*, the number of resolutions m;
$prob_1, prob_2, \cdots, prob_m = \frac{1}{m}, \frac{1}{m}, \cdots, \frac{1}{m}$;
for $i = 1$ **to** *iter* **do**
  Select the resolution r according to $prob_1, \cdots, prob_m$;
  Train FPN by descending gradient: $\nabla_{\theta_{FPN}} \mathcal{L}_{loss}(Y_{X,r})$
end for

5. Discussion about the limitation

As shown in Fig. 5-(b), our method shows relatively inferior performances at resolutions below 64. FPN may fail to cover the whole shapes at low resolutions when the initial randomly sampled points miss some thin structures. However, FPN is still meaningful as it can significantly improve the sampling efficiency and task-oriented performances at relatively high resolutions as shown in Sections 4.2 and 4.4. We will try more to overcome the limitation in the future work.

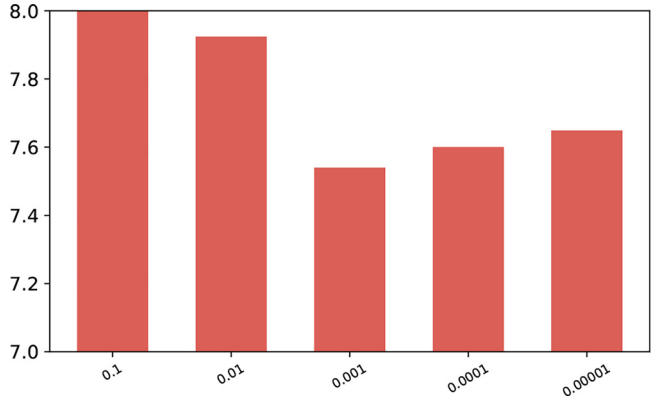

Fig. 10. The influence of range constraint.

6. Conclusion

In this work, we propose a fast point clouds sampling network (FPN). We use the widely used random sampling to generate initial points and drive them to more appropriate positions with the network. The sampling resolution of FPN is flexible, which supports sampling different resolution points with a same network. We introduce hybrid training strategy to enhance the adaptability to different resolutions by randomly selecting different resolutions during training. The experiments on point cloud reconstruction and recognition have demonstrated that FPN can achieve better performances than most existing sampling strategies, when it is much more efficient than many well-performed methods such as FPS, S-Net [5] and SamNet [4].

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

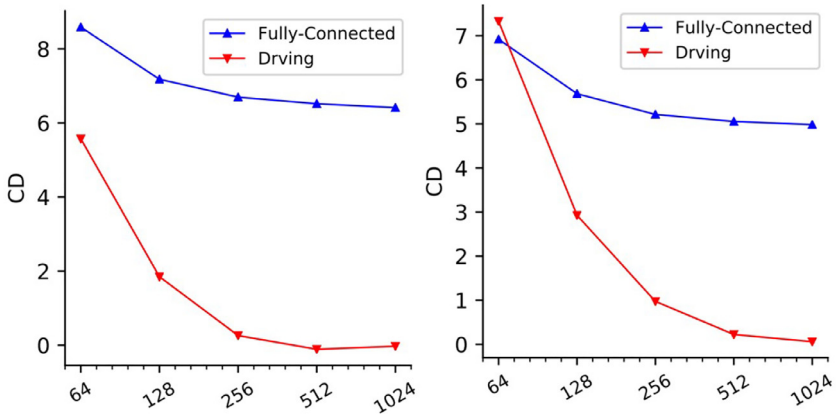

Fig. 9. Comparison of different sampling network organization. (a) and (b) denote reconstruction errors measured on ModelNet10 and ModelNet40.

Acknowledgment

We thank all reviewers and the editor for excellent contributions. This work is supported by the Key Research and Development Project of Zhejiang Province under Grant 2021C01035.

References

[1] C.R. Qi, L. Yi, H. Su, L.J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: Advances in neural information processing systems, 2017, pp. 5099–5108.
[2] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, A. Markham, Randla-net: Efficient semantic segmentation of large-scale point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11108–11117.
[3] C.R. Qi, O. Litany, K. He, L.J. Guibas, Deep hough voting for 3d object detection in point clouds, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9277–9286.
[4] I. Lang, A. Manor, S. Avidan, Samplenet: Differentiable point cloud sampling, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 7578–7588.
[5] O. Dovrat, I. Lang, S. Avidan, Learning to sample, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2760–2769.
[6] A. Dai, C. Ruizhongtai Qi, M. Nießner, Shape completion using 3d-encoder-predictor cnns and shape synthesis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5868–5877.
[7] X. Han, Z. Li, H. Huang, E. Kalogerakis, Y. Yu, High-resolution shape completion using deep neural networks for global structure and local geometry inference, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 85–93.
[8] D. Li, T. Shao, H. Wu, K. Zhou, Shape completion from a single rgbd image, IEEE transactions on visualization and computer graphics 23 (7) (2016) 1809–1822.
[9] C.R. Qi, H. Su, K. Mo, L.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, pp. 652–660.
[10] B.-S. Hua, M.-K. Tran, S.-K. Yeung, Pointwise convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 984–993.
[11] T. Le, Y. Duan, Pointgrid: A deep network for 3d shape understanding, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9204–9214.
[12] Y. Chen, S. Liu, X. Shen, J. Jia, Fast point r-cnn, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 9775–9784.
[13] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, J. Kautz, Splatnet: Sparse lattice networks for point cloud processing, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2530–2539.
[14] Z. Han, X. Wang, Y.-S. Liu, M. Zwicker, Multi-angle point cloud-vae: unsupervised feature learning for 3d point clouds from multiple angles by joint self-reconstruction and half-to-half prediction, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE, 2019, pp. 10441–10450.
[15] C. Wang, B. Samari, K. Siddiqi, Local spectral graph convolution for point set feature learning, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 52–66.
[16] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, B. Hariharan, Pointflow: 3d point cloud generation with continuous normalizing flows, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 4541–4550.
[17] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, P.-A. Heng, Pu-net: Point cloud upsampling network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2790–2799.
[18] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, P.-A. Heng, Pu-gan: a point cloud upsampling adversarial network, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 7203–7212.
[19] Y. Wang, Y. Sun, Z. Liu, S.E. Sarma, M.M. Bronstein, J.M. Solomon, Dynamic graph cnn for learning on point clouds, Acm Transactions On Graphics (tog) 38 (5) (2019) 1–12.
[20] W. Wu, Z. Qi, L. Fuxin, Pointconv: Deep convolutional networks on 3d point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9621–9630.
[21] H. Thomas, C.R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, L.J. Guibas, Kpconv: Flexible and deformable convolution for point clouds, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6411–6420.
[22] H. Fan, H. Su, L.J. Guibas, 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, 2017, pp. 605–613.
[23] A.X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al., Shapenet: An information-rich 3d model repository, arXiv preprint arXiv:1512.03012 (2015).
[24] Y. Yang, C. Feng, Y. Shen, D. Tian, Foldingnet: Point cloud auto-encoder via deep grid deformation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 206–215.
[25] P. Achlioptas, O. Diamanti, I. Mitliagkas, L. Guibas, Learning representations and generative models for 3d point clouds, in: International conference on machine learning, PMLR, 2018, pp. 40–49.
[26] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3d shapenets: A deep representation for volumetric shapes, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1912–1920.
[27] Y. Rao, J. Lu, J. Zhou, Global-local bidirectional reasoning for unsupervised representation learning of 3d point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5376–5385.
[28] T.F. Gonzalez, Clustering to minimize the maximum intercluster distance, Theoretical computer science 38 (1985) 293–306.