7 Appendix

7.1 Experimental Settings

Task Networks Similar to SampleNet [6], we adopt PointNet for classification [7], Point Cloud Autoencoder (PCAE) for reconstruction [1], and PCRNet for registration [8]. For the classification and reconstruction tasks, PointNet and PCAE are trained with the same settings as reported by their original papers. For the registration task, Sarode et al. [8] trained the PCRNet with the Chamfer distance between template point cloud and registered point cloud; we follow SampleNet and add a regression loss (besides the Chamfer distance) to train the PCRNet. These pre-trained networks are treated as the task networks for their specific applications, whose parameters are fixed during the training of APSNet.

APSNet The feature extract component of APSNet follows the design of PointNet [7]. It contains a sequence of 1×1 convolution layers, followed by a symmetric global pooling layer to generate a global feature vector, which is then used as the initial state of LSTM for sampling. Each convolution layer includes a batch normalization layer [4] and a ReLU activation function. A 2-layer LSTM [3] with 128 recurrent units in each layer is used to generate samples autoregressively.

We consider two variants of APSNet: (1) *APSNet*, and (2) *APSNet-KD*, while the former refers to the supervised training of APSNet and the latter refers to the self-supervised training of APSNet with knowledge distillation. A trained APSNet generates point cloud $\boldsymbol{Q}$ that isn't a subset of original input point cloud $\boldsymbol{P}$, but the generated $\boldsymbol{Q}$ can be converted to $\boldsymbol{Q}^*$ by a matching process as discussed in Sec. 3.1. Therefore, we further distinguish them as *APSNet-G* and *APSNet-M*, respectively.

SampleNet [6] also generates point cloud $\boldsymbol{Q}$, which is converted to $\boldsymbol{Q}^*$ by the matching process. Similarly, we denote them as *SampleNet-G* and *SampleNet-M*, respectively. In our experiments, we compare the performances of all these variants. However, we would like to emphasize that the default SampleNet is *SampleNet-M*, while the default APSNet is *APSNet-G* since APSNet-G yields the best predictive performance without an expensive matching process as we will demonstrate in the experiments.

Implementation We tune the performance of APSNet based on the hyperparameters provided by SampleNet [6], and set $\beta = 1$, $\gamma = 1$ and $\delta = 0$. We use the Adam optimizer [5] with the batch size of 128 for all the experiments. Learning rate is set to (0.01, 0.001, 0.0005), and λ of the total loss (9) is set (30, 0.01, 0.01) for classification, registration, and reconstruction tasks, respectively. Each experiment is trained for 400 epochs with a learning rate decay of 0.7 at every 20 epochs.

Since our code is PyTorch-based, we convert the official TensorFlow code of SampleNet[1] to PyTorch for a fair comparison. We found that our PyTorch implementation achieves better performances than the official TensorFlow version in most of our experiments. For reproducibility, our source code is also provided as a part of the supplementary material. All our experiments are performed on Nvidia RTX GPUs.

7.2 Reconstruction

The reconstruction task is evaluated with point clouds of 2048 points, sampled from the ShapeNet Core55 dataset [2]. We choose the four shape classes that have the largest number

[1] https://github.com/itailang/SampleNet

Table 1: The normalized reconstruction errors of five sampling methods with different sample sizes m on the ShapeNet Core55 dataset. M* denotes the original results from the SampleNet paper [6]. The lower, the better.

	RS	FPS	SampleNet			APSNet		APSNet-KD	
m			G	M	M*	G	M	G	M
8	21.85	12.79	5.29	5.48	-	**4.27**	4.59	4.69	4.98
16	13.47	7.25	2.78	2.89	-	**2.51**	2.62	2.57	2.67
32	8.16	3.84	1.68	1.71	2.32	1.54	1.59	**1.47**	1.52
64	4.54	2.23	1.32	1.27	1.33	**1.07**	1.11	1.12	1.14

of examples: Table, Car, Chair, and Airplane. Each class is split to a 85%, 5%, 10% partition for training, validation and test. The task network, in this case, is the Point Cloud Autoencoder (PCAE) for reconstruction [1]. We evaluate the reconstruction performance with the normalized reconstruction error (NRE):

$$\mathsf{NRE}_{\mathsf{CD}}(\boldsymbol{Q},\boldsymbol{P}) = \frac{\mathsf{CD}(\boldsymbol{P},T(\boldsymbol{Q}))}{\mathsf{CD}(\boldsymbol{P},T(\boldsymbol{P}))}, \tag{1}$$

where CD is the Chamfer distance [1] between two point clouds. Apparently, the values of $\mathsf{NRE}_{\mathsf{CD}}$ are lower bounded by 1, and the smaller, the better.

Table 1 reports the reconstruction results of all the five sampling methods considered. Similar to the results of classification, (1) SampleNet and APSNet outperform RS and FPS by a large margin. (2) SampleNet-M relies on the matching process to replace the redundant samples by FPS to improve its performance over SampleNet-G. (3) In contrast, APSNet-G outperforms APSNet-M consistently without the extra matching process. (4) APSNet-KD again achieves a very competitive result to APSNet. (5) Comparing APSNet-G and SampleNet-M (the best defaults for both algorithms), APSNet outperforms SampleNet consistently by a notable margin.

To investigate why APSNet outperforms SampleNet in the task of reconstruction, we visualize the sampled points and the reconstructed point clouds of both algorithms in Fig. 1. As can be seen, SampleNet focuses more on the main body of airplane and samples some uninformative and symmetric points for reconstruction. In contrast, APSNet focuses more on the outline of the airplane without losing details, which are critical for the reconstruction. This observation is more pronounced when sample size is small, such as $m = 8$. As shown in Fig. 1(a) and (b), SampleNet fails to sample a point at the tail of the airplane such that the reconstructed point cloud cannot recover the tail. In comparison, APSNet samples two important points at the tail and ignores the symmetric one on the other side of the tail, and therefore is able to reconstruct the tail precisely. One the other hand, SampleNet samples two symmetric points on the wing, which are likely redundant information for the reconstruction. Overall, the sampled points from APSNet are more reasonable than those of SampleNet from human's perspective,

The effectiveness of different loss components The sampling loss (7) encourages the sampled points in $\boldsymbol{Q}$ to be close to those of $\boldsymbol{P}$ and also have a maximal coverage w.r.t. the original point cloud $\boldsymbol{P}$. We found that this sampling loss provides an important prior knowledge for sampling, and is critical for APSNet to achieve a good performance. In addition, the sampling loss (7) is more generic than the Chamfer distance since when $\beta = 0$, $\gamma = 1$ and $\delta = 0$ it degenerates to the Chamfer distance. The limitation is that we now have more hyperparameters to tune. Table 2 reports the ablation study of the sampling loss (7) for APSNet-G

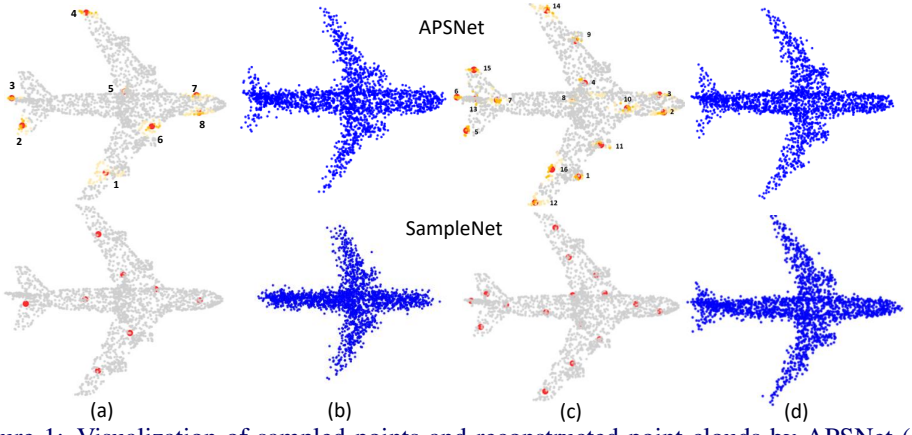

Figure 1: Visualization of sampled points and reconstructed point clouds by APSNet (1st row) and SampleNet (2nd row). The red dots are the sampled points; the highlighted yellow regions in APSNet results are points with high attention scores and the number specify the order of sampled points. (a) Sampled points when $m = 8$; (b) Reconstruction when $m = 8$, NRE(APSNet)=2.55, NRE(SampleNet)=5.20; (c) Sampled points when $m = 16$; (d) Reconstruction when $m = 16$, NRE(APSNet)=1.57, NRE(SampleNet)=2.34.

on the reconstruction task. It shows that when $L_m(\boldsymbol{Q},\boldsymbol{P})$ and $L_a(\boldsymbol{P},\boldsymbol{Q})$ enabled (i.e., $\beta = 1$ and $\gamma = 1$), APSNet-G reaches the best results in almost all settings.

Table 2: Ablation study of the sampling loss (7) for APSNet-G on the reconstruction task. * denotes the best results when $\beta = 1$, $\gamma = 1$ and $\delta = 0$.

	$\beta = 0$	$\gamma = 0$	*
8	4.63	4.54	4.27
16	3.07	3.44	2.51
32	1.67	1.49	1.54
64	1.13	1.28	1.07

Inference time comparison We further evaluate the inference times of different sampling methods in the task of reconstruction. The results are reported in Table 3, where SampleNet-M and APSNet-G are the main algorithms to be compared since they are the best defaults. It can be observed that when sample size m increases, the inference times of both SampleNet-M and APSNet-G increase, while SampleNet-G requires roughly a constant time for sampling. This is because SampleNet-G leverages an MLP generator to generate all m samples in one shot; for the problem size considered, one GPU is able to utilize its on-board parallel resources to process different sample sizes in roughly the same time. However, as observed in the experiments above and also proposed by SampleNet [8], SampletNet relies on the matching process to improve its performance, while matching is the most expensive operation in SampleNet, leading to a dramatic increase of inference-time for SampleNet-M. By contrast, due to the autoregressive model of our method, APSNet generates samples sequentially by an LSTM which results in a linear increase of inference time as m increases. However, APSNet does not need an expensive matching process for its best performance. Therefore, besides the improved sample quality, APSNet also outperforms SampleNet in terms of inference time.

Table 3: Inference time comparison of three sampling methods with different sample size m. The time is reported in millisecond. * denotes the best default recommended by each paper.

m	32	128	256	512
SampleNet-G	7.63	7.54	7.79	7.94
SampleNet-M*	44.33	135.23	261.47	515.30
APSNet-G*	9.21	12.84	17.68	27.48
APSNet-M	45.91	139.83	269.40	525.38

7.3 Registration

The task of registration aims to align two point clouds by predicting rigid transformations (e.g., rotation and translation) between them. To save memory and computation power, the registration is conducted on the key points that are sampled from the original point clouds. We follow the work of PCRNet [8] to construct a point cloud registration network (the task network), and train PCRNet on the point clouds of 1,024 points of the Car category from ModelNet40. Following the settings in SampleNet, 4,925 pairs of source and template point clouds are generated for training, where a template is rotated by three random Euler angles in the range of $[-45°, 45°]$ to obtain the source. An additional 100 source-template pairs are generated from the test split for evaluation. The mean rotation error (MRE) between the predicted rotations and ground-truth rotations is used as the evaluation metric.

Table 4 reports the performances of five sampling methods for registration. Similar to the results on classification and reconstruction, APSNet outperforms SampleNet consistently by a notable margin, and achieves the state-of-the-art results in this task. Without leveraging labeled training data, APSNet-KD again demonstrates an impressive performance that is close to supervised APSNet.

Table 4: The mean rotation errors of five sampling methods with different sample sizes m on the ModelNet40 dataset for registration. M* denotes the original results from the SampleNet paper [8]. The lower, the better.

	RS	FPS	SampleNet			APSNet		APSNet-KD	
m			G	M	M*	G	M	G	M
8	63.37	31.44	9.72	8.27	10.51	**5.47**	9.40	5.93	10.51
16	43.89	20.34	12.14	7.45	8.21	**4.50**	7.18	5.01	7.07
32	27.06	12.97	10.81	6.13	5.94	**4.37**	5.82	4.56	6.07
64	16.88	7.89	10.93	5.38	5.31	**4.42**	6.34	4.49	4.97

Visualization of Attention Coefficients For the task of registration, we further visualize the evolution of attention coefficients during the training process. Specifically, we monitor the attention coefficients Eq. (3) when generating a point at a specific time step t (the t-th sample) from a given point cloud of 1024 points. Figure 2 visualizes the evolution of attention coefficients over 400 training epochs. At beginning of the training, the sampler cannot decide which point from the point cloud is the most important one to sample, manifested by the dense cluttered coefficients. As the training proceeds, the attention coefficients become sparser with peak values on 2-3 points. Further, these attention coefficients are stablized in the late training epochs and consistently concentrate on a few the same points, demonstrating the training stability of APSNet.

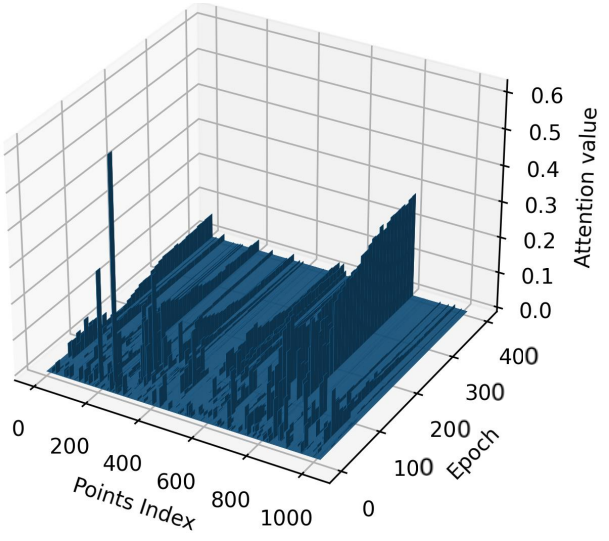

Figure 2: Evolution of the attention coefficients of APSNet when generating the t-th sample. As the training proceeds, the coefficients become sparser with peak values on a few points.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning Representations and Generative Models For 3D Point Clouds. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 40–49, 2018.

[2] Angel X. Chang, Thomas Funkhouser, Leonidas J. 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. *arXiv preprint arXiv:1512.03012*, 2015.

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[4] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[6] 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 (CVPR)*, pages 7578–7588, 2020.

[7] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017.

[8] Vinit Sarode, Xueqian Li, Hunter Goforth, Rangaprasad Arun Aoki, Yasuhiro Srivatsan, Simon Lucey, and Howie Choset. PCRNet: Point Cloud Registration Network using PointNet Encoding. *arXiv preprint arXiv:1908.07906*, 2019.