Task-Oriented Compact Representation of 3D Point Clouds via A Matrix Optimization-Driven Network

Yue Qian, Junhui Hou, *Senior Member, IEEE,*, Qijian Zhang, Yiming Zeng, Sam Kwong, *Fellow, IEEE,* and Ying He

***Abstract*—This paper explores the task-oriented compact representation of 3D point clouds, which should maintain the performance of subsequent applications applied to such compact point clouds as much as possible. Designing from the perspective of matrix optimization, we propose MOPS-Net, a novel deep learning-based method that is distinguishable from existing approaches due to its interpretability and flexibility. The matrix optimization problem is challenging due to the *discrete* and *combinatorial* nature of the sampling matrix. Therefore, we tackle the challenges by relaxing the binary constraint of the sampling matrix and formulating a constrained and differentiable optimization problem. We then design a deep neural network to mimic the matrix optimization by exploring both the local and global structures of the input data. MOPS-Net can be end-to-end trained with a task network and is permutation-invariant, making it robust to the input. We also extend MOPS-Net such that a single network after one-time training is capable of handling arbitrary downsampling ratios. Extensive experimental results show that MOPS-Net can achieve favorable performance against state-of-the-art deep learning-based methods over various tasks, including classification, reconstruction, and registration. Besides, we validate the robustness of MOPS-Net on noisy data.**

***Index Terms*—Point cloud, Sampling, Optimization, Deep learning, Classification, Reconstruction, Registration.**

I. Introduction

WITH recent advances in three-dimensional (3D) sensing technology (e.g., LiDAR scanning devices), 3D point clouds can be easily obtained. Compared with other 3D representations such as multi-view images, voxel grids and polygonal meshes, point clouds are a raw 3D representation, containing only 3D samples which are located on the scanned surface. Powered by deep learning techniques, the performance of many point cloud applications, such as classification, segmentation and reconstruction, has been improved significantly in recent years. However, processing large-scale and/or dense 3D point clouds is still challenging due to the high cost of computation, storage, and communication load.

3D point clouds in a compact representation can help reduce information redundancy, thereby improving the speed of the downstream applications and saving storage space and transmission bandwidth. To compactly represent 3D point cloud, existing works in the field of point cloud compression try to establish the point relation via a tree-based structure [1], [2], [3], [4], a projection to 2D images [5], [6], and compression of the intermediate feature code [7], [8], [9]. These compression works aim to reduce the data volume of an input 3D point cloud with introducing as little quality degradation as possible.

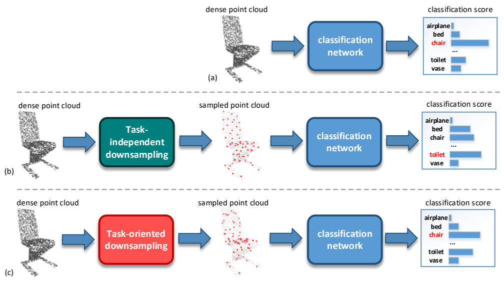


Fig. 1. Illustrations of task-independent and task-oriented point cloud downsampling using classification as an example. (a) A deep learning-based point cloud classifier is trained on dense point clouds. (b) Traditional downsampling methods, such as FPS, generate sparse point clouds without considering the nature of the task, hereby may compromise the performance of the classifier in (a) significantly. (c) The classification-oriented downsampling method produces sparse point clouds that maintain the performance of the classifier in (a). To develop such classification-oriented downsampling, we take both the geometry of the input shape and the performance of the classifier into account.

Besides, downsampling is also a popular and effective technique to achieve the compact representation of 3D point clouds. In contrast to point cloud compression, the compact representation created by downsampling is still in the 3D point cloud space. The traditional downsampling approaches such as farthest point sampling (FPS) [10] and Poisson disk sampling (PDS) [11] iteratively generate uniformly distributed samples on the input shape, and thus they can preserve the geometry well. Such sampling methods, however, focus on reducing the geometry loss only and are completely independent of the downstream applications. As a result, under a limited sampling budget, traditional approaches cannot adaptively preserve the most informative points according to specific task properties, which causes sub-optimal downsampling efficacy and degraded task performance.

An alternative way for compactly representing 3D point clouds is to generate samples that optimize the performance of a particular task, i.e., the resulting sparse point clouds will maintain the task performance as much as possible. Due to the task-centric nature, we call it *task-oriented* compact

This work was supported in part by the Hong Kong Research Grants Council under grants 11202320 and 11219422, and in part by the Natural Science Foundation of China under Grants 61871342. (*Corresponding author: Junhui Hou*)

Y. Qian, J. Hou, Q. Zhang, Y. Zeng, and S. Kwong are with the Department of Computer Science, City University of Hong Kong, Hong Kong, and also with the City University of Hong Kong Shenzhen Research Institute, Shenzhen 518057, China. Email: yueqian4-c@my.cityu.edu.hk; jh.hou@cityu.edu.hk; ym.zeng@my.cityu.edu.hk; qijizhang3@cityu.edu.hk; cssamk@cityu.edu.hk

Y. He is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore, 639798. Email: yhe@ntu.edu.hk

representation. Moreover, an effective downsampling method should allow the user to freely specify the downsampling ratio to balance the task performance and computation efficiency. As the deep learning technologies have proven effective in point cloud classification [12], [13], [14], [15], [16], segmentation [17], [18], [19], [20], [21], registration [22], [23], [24], [25] and compression [26], [27], [28], it is highly desired to combine downsampling methods with the deep neural networks. However, extending the existing network architectures to point cloud downsampling is non-trivial since the point selection process is discrete and non-differentiable. Recently, Dovrat et. al. pioneered a deep learning approach called S-Net [29], which takes downsampling as a generative task and uses the extracted global features to generate samples. S-Net is flexible in that it can be combined with task-specific networks to produce an end-to-end network trained by a joint loss. Thanks to its task-oriented nature, S-Net outperforms FPS in various applications. Hereafter, Lang *et al.* [30] improved S-Net by introducing an additional projection module to encourage the generated points closer to original point clouds, learning SampleNet. However, as S-Net and SampleNet solely rely on global features in their generative processes, they can not utilize the point-wise high-dimensional local features, which limits the quality of synthesized points.

In this paper, we propose a novel deep learning approach, namely MOPS-Net, to obtain the task-oriented compact representation of 3D point clouds. In contrast to the existing methods, we propose MOPS-Net from the matrix optimization perspective. Viewing downsampling as a selection process, we first formulate a discrete and combinatorial optimization problem. As it is difficult to solve the 0-1 integer program directly, we relax the integer constraint for each variable, in which a constrained and differentiable matrix optimization problem with an implicit objective function is formulated. We then design a deep neural network architecture to mimic the matrix optimization problem by exploring both the local and global structures of the input data. With a task network, MOPS-Net can be end-to-end trained. MOPS-Net is permutation-invariant, making it robust to input data. Furthermore, we extend our method to handle arbitrary downsampling ratios by constraining the invoking columns of the soft sampling matrix. Our extension allows a single network, trained only once, to be utilized for downsampling tasks involving varying ratios. Extensive results show that MOPS-Net achieves much better performance than state-of-the-art deep learning-based methods and traditional methods in various tasks, including point cloud classification, reconstruction, and registration. We also provide comprehensive ablation studies and analysis on model robustness and time efficiency.

The main contributions of this paper are summarized as follows:

- we explicitly formulate the problem of the task-oriented compact representation of 3D point clouds from the matrix optimization perspective;
- we propose an interpretable deep learning-based method named MOPS-Net, which mimics the above formulation, making it different from the existing deep learning-based methods that directly regress the downsampled point sets.
- we extend MOPS-Net and propose FMOPS-Net, which is capable of handling arbitrary downsampling ratios after only one-time training; and
- we propose a new compact deep-learning framework for large-scale point cloud reconstruction, which enables the validation of the scalability of MOPS-Net on relatively large-scale point clouds; and
- we conduct various experiments and comprehensive ablation studies to demonstrate the advantages of our methods over state-of-the-art methods.

The rest of this paper is organized as follows. Section II briefly reviews point cloud compression works, traditional point cloud downsampling methods and recent deep learning techniques for 3D point clouds. In Section III, we formulate the problem of the task-oriented compact representation as a constrained and differentiable optimization problem, followed by an end-to-end deep neural network that mimics the resulting optimization in Section IV. Section V presents extensive experimental results, comparisons with the state-of-the-art, as well as comprehensive robustness tests and ablation studies. Section VI finally concludes this paper.

II. Related work

A. Point Cloud Compression

One kind of traditional approaches attempts to reorganize unstructured 3D point clouds into a memory-efficient data structure such as Octree [1], [2], [3], [4] and region-wise clustering [31]. Another kind of popular traditional point cloud compression methods is based on 3D-to-2D mapping [5], [6]. Specifically, they project 3D objects to multiple 2D images from various viewpoints and then adopt the mature image and video codecs to compress the resulting 2D images. Jia *et al.* [32] further improved the traditional 3D-to-2D projection method by introducing a CNN to compress the 2D occupancy map. We also refer the reader to [6] [33] for the comprehensive review of 3D point cloud compression. Inspired by the success of the deep learning-based image compression [34], [35], some deep learning-based point cloud compression frameworks have recently emerged [7], [8], [9], [36], [37]. Basically, these methods construct auto-encoder structures that allow the encoders to convert an input point cloud to a feature code. The extracted feature is further quantified and entropy coded into a bitstream.

B. Traditional Downsampling Methods

The traditional methods, such as farthest point sampling (FPS) and Poisson disk sampling (PDS), generate samples in an iterative manner. Starting from a random sample, FPS repeatedly places the next sample point in the center of the least-sampled area. Using efficient geodesic distance computation tools (such as the fast marching method [38]), FPS generates m samples on a n-vertex mesh in $\sum_{i=1}^{m} f(\frac{n}{i}) = O(n \log n)$ time, where $f(x) = O(x \log x)$. FPS is easy to implement and becomes popular in designing neural networks that aggregate local features [12], [14], [39]. Poisson disk sampling produces samples that are tightly packed, but no closer to each other than a specified minimum distance, resulting in a more uniform distribution than FPS. There are efficient implementations of

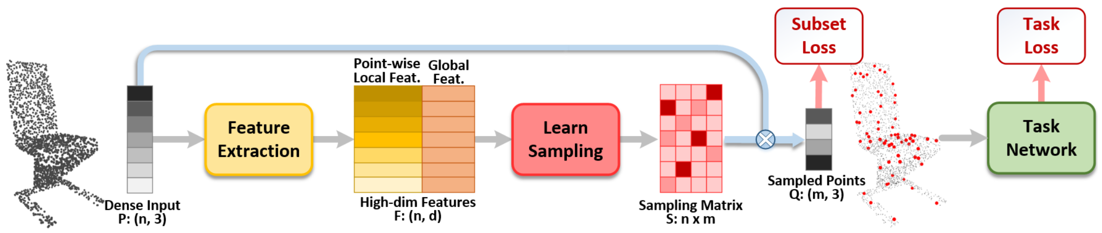


Fig. 2. MOPS-Net is a matrix optimization-driven deep learning method for task-oriented 3D point cloud downsampling. It first extracts high-dimensional features, which contain both global and local geometric information, from point coordinates (Sec. IV-B). It then utilizes the features to learn a differentiable sampling matrix, which is multiplied by the input dense point cloud to obtain the sampled points (Sec IV-C). Finally, it feeds the downsampled sparse points to a task network. The whole network is trained by jointly optimizing the task loss and the subset loss (Sec. IV-E).

PDS with linear time complexity in Euclidean spaces [40], [41]. However, it is expensive to generate Poisson disk samples on curved surfaces due to frequent computation of geodesic distances [11]. Voxelization is also a commonly used technique to downsample or resample point clouds, which quantizes point clouds into regular voxels in 3D space with predefined resolution. Compared with FPS and PDS, voxelization is more efficient due to its non-iterative manner. However, voxelization often suffers from quantization error and cannot yield results that are the exact subsets of the input. Moreover, the traditional approaches focus only on preserving the geometry of the input shape and they do not consider the downstream tasks at all. Despite their simple implementation, a traditional downsampling strategy, such as FPS, has been widely applied to various point cloud processing methods, including deep learning-based methods [12], [42], [43].

C. Deep Learning for 3D Point Clouds

Due to the irregular and unordered nature of point clouds, the widely used convolutional neural networks (CNNs) on 2D images/videos [44], [45], [46] cannot be applied directly. PointNet, proposed by Qi *et al.* [13], maps a 3D point to a high dimensional space by point-wise multi-layer perceptrons (MLPs) and aggregates global features by a symmetric function, named max-pooling. As the first deep neural network that works for 3D raw points without projecting or parameterizing them to regular domains, PointNet quickly gained popularity and was successfully used as the fundamental feature extraction for point clouds. However, PointNet processes the points individually and does not consider the spatial relation among points. The follow-up works, such as PointNet++ [12], DGCNN [16] and PointCNN [14], improve PointNet by taking local geometry into account.

Inspired by the success of PointNet on classification, many other point cloud applications were studied in recent years, such as retrieval [47], [48], segmentation [17], [18], [19], [20], [49], reconstruction [50], [51], [52], registration [53], [25], [54], [55], [56], object detection [57], [58], [59], [60], [61] and hand pose estimation [62], just name a few. Although they have different problem settings, these networks can be combined with the point cloud downsampling network and jointly trained. In this paper, we evaluate the performance of the proposed downsampling framework on classification, registration, and reconstruction.

Opposite to downsampling, point cloud upsampling [63], [64], [65], [66] has also been investigated recently. Upsampling can be treated as either a 3D version of image superresolution, or the inverse process of downsampling. Despite the common word "sampling", the two tasks are completely different. Upsampling, as a generative task, requires informative feature expansion and can be trained by ground truth dense point clouds. In contrast, point cloud downsampling is close to feature selection, where a differentiable end-to-end framework should be carefully designed. Moreover, due to a lack of ground truth, the downsampled points should be learned to optimize a specific task loss.

D. Deep Learning-based Point Cloud Downsampling

It is an emerging topic, on which there are only a few works. Nezhadary *et al.* [67] proposed to use critical points invoked in max-pooling as sampled points. In order to improve classification accuracy, Yang [68] adopted a Gumbel softmax layer to integrate high-level features. Recently, Dovrat *et al.* [29] proposed a data-driven point cloud downsampling framework named S-Net. After the point-wise feature extraction by PointNet, a global feature was obtained by the max-pooling operation. Then 3D coordinates of fewer points were regressed by fully-connected layers. Followed by a pre-trained task network, S-Net can be trained end-to-end. Lang *et al.* [30] proposed SampleNet, which improves S-Net by introducing a soft projection module that adopts an annealing schedule to encourage generated points to be close to original points. However, both S-Net and SampleNet regress coordinates from global features directly and do not consider spatial correlation among sampled points, which plays an important role in downsampling, since spatially close points have the tendency to be represented by the same downsampled point. Recently, Liu *et al.* [69] proposed APSNet, which can adaptively adjust the resolution of point clouds for the purpose of action recognition. However, APSNet performs downsampling via FPS and requires four pre-defined levels of resolution. In sharp contrast to current deep learning-based downsampling strategies, we propose a novel framework by exploring the local geometry of the input data from a perspective of matrix optimization, making it applicable to diverse downstream applications and granting it the flexibility to accommodate arbitrary downsampling ratios.

III. Problem Formulation

In this section, we explicitly formulate the problem of the task-oriented compact representation of 3D point clouds from the matrix optimization perspective.

Denote by $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^n$ a dense point cloud with n points. $\mathbf{p}_i = \{x_i, y_i, z_i\}$ is the 3D Cartesian coordinates. Let $\mathcal{Q} = \{\mathbf{q}_i \in \mathbb{R}^3\}_{i=1}^m$ be the downsampled sparse point cloud with m $(< n)$ points, which is a subset of $\mathcal{P}$, i.e., $\mathcal{Q} \subset \mathcal{P}$.

As aforementioned, we consider a task-oriented sampling process. That is, given $\mathcal{P}$, we compute the downsampled point cloud $\mathcal{Q}$ that maintains comparable or at least does not compromise the performance of the subsequent tasks too much, e.g., classification, etc. Such a compact representation is beneficial to computation, storage, and transmission.

Mathematically speaking, the problem can be formulated as

$$\min_{\mathcal{Q}} L_{task}(\mathcal{Q}) \quad s.t. \quad \mathcal{Q} \subset \mathcal{P}, \tag{1}$$

where $L_{task}(\cdot)$ indicates the task-tailored loss, whose explicit form will be discussed in Section IV-E. We rewrite Eq. (1) in a more explicit manner

$$\begin{aligned} &\min_{\mathbf{S}} L_{task}(\mathbf{Q}) \\ &s.t.\ \mathbf{Q} = \mathbf{S}^{\mathsf{T}}\mathbf{P},\ \mathbf{1}_n^{\mathsf{T}}\mathbf{S} = \mathbf{1}_m^{\mathsf{T}},\ \mathbf{S}^{\mathsf{T}}\mathbf{S} = \mathbf{I}_m,\ s_{i,j} \in \{0,1\}, \end{aligned} \tag{2}$$

where $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n]^{\mathsf{T}} \in \mathbb{R}^{n\times 3}$ and $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m]^{\mathsf{T}} \in \mathbb{R}^{m\times 3}$ are the matrix representations of $\mathcal{P}$ and $\mathcal{Q}$, respectively, constructed by stacking each point as a column in an *unodered* manner[1]; $s_{i,j}$ is the (i, j)-th entry of $\mathbf{S} \in \mathbb{R}^{n\times m}$; $\mathbf{1}_n = [1, \ldots, 1]^{\mathsf{T}} \in \mathbb{R}^{n\times 1}$ is the column vector with all elements equal to 1; and $\mathbf{I}_m$ refers to the identity matrix of size $m \times m$. The constraints force $\mathbf{S}$ to be an ideal sampling matrix, i.e., $\mathbf{S}$ only contains m columns of a permutation matrix of size $n \times n$. More precisely, the orthogonal constraint is used to avoid repeated columns (indicating that an identical point is selected multiple times) in $\mathbf{S}$.

The challenge for solving Eq. (2) comes from the discrete and binary characteristics of matrix $\mathbf{S}$. To tackle this challenge, we relax the binary constraints Eq. (2) in a soft and continuous manner, i.e., the elements of $\mathbf{S}$ are continuous, ranging between 0 and 1. The relaxed variables indicate the probabilities of the corresponding points that will be sampled. After this relaxation, the resulting points in $\mathcal{Q}$ may not be the subset of $\mathcal{P}$. To mitigate this effect for a meaningful sampling process, we further introduce a metric $L_{subset}(\cdot,\cdot)$ to quantitatively measure the distance between two point clouds, and minimizing $L_{subset}(\mathbf{P}, \mathbf{Q})$ will promote $\mathbf{Q}$ to be a subset of $\mathbf{P}$ as much as possible. We will explain the explicit form of $L_{subset}(\cdot,\cdot)$ in Section IV-E.

We finally express the *relaxed* and *continuous* optimization problem for task-oriented point cloud downsampling as

$$\begin{aligned} &\min_{\mathbf{S}} L_{task}(\mathbf{Q}) + \alpha L_{subset}(\mathbf{P}, \mathbf{Q}) \\ &s.t.\ \mathbf{Q} = \mathbf{S}^{\mathsf{T}}\mathbf{P},\ \mathbf{S} \geq 0,\ \mathbf{1}_n^{\mathsf{T}}\mathbf{S} = \mathbf{1}_m^{\mathsf{T}},\ \|\mathbf{S}^{\mathsf{T}}\mathbf{S} - \mathbf{I}_m\|_F < \epsilon, \end{aligned} \tag{3}$$

[1]Note $\mathcal{P}$ (resp. $\mathcal{Q}$) and $\mathbf{P}$ (resp. $\mathbf{Q}$) stand for the same data but in different forms, and there is no specific requirement on the order when stacking the points. The notations are used interchangeably in the paper.

where $\|\cdot\|_F$ is the Frobenius norm of a matrix, $\epsilon > 0$ is a threshold, and $\alpha > 0$ is the penalty parameter to balance the two terms.

Remarks. Although the optimization of the matrix $\mathbf{S}$ in deep learning frameworks requires that it be non-binary, we introduce a temperature annealing mechanism to enforce it to be close to a binary matrix, as demonstrated in Section IV-C. However, the non-binary matrix $\mathbf{S}$ still generates a point cloud that is not necessarily a subset of the input $\mathcal{Q} \not\subset \mathcal{P}$. This can be explained from the perspective of geometry processing. When presenting an object using two point clouds of different resolutions, the one with fewer points is generally not fully overlapping with the larger one since we have to re-distribute the points in order to preserve the geometry. Although $\mathcal{Q} \not\subset \mathcal{P}$, the objective function penalizes the points that are away from the input shape. Therefore, the points of $\mathcal{Q}$ are either on or close to the underlying object surface. If the subsequent task requires $\mathcal{Q} \subset \mathcal{P}$, we can assign each point of $\mathcal{Q}$ the closest point in the input data. In Section V, we will quantitatively analyze the effect of these post-processing operations.

IV. Proposed Method

This section presents MOPS-Net, a novel end-to-end deep neural network that mimics the formulated optimization problem in Eq. (3) for the task-oriented compact representation of 3D point clouds.

A. Overview

Figure 2 illustrates the flowchart of MOPS-Net. Given a point cloud $\mathcal{P}$ and a pre-trained task network, MOPS-Net uses a feature extraction module to encode each point with high dimensional and informative features by exploring both the local and the global structures of $\mathcal{P}$ (Section IV-B). Based on the high dimensional features, MOPS-Net estimates a differentiable sampling matrix $\mathbf{S}$ under the guidance of the constraints in Eq. (3). Multiplying the learned sampling matrix to original dense point clouds, we can obtain the sampled points efficiently (Section IV-D). Together with a fixed task network, MOPS-Net can be end-to-end trained with a joint loss (Section IV-E). Such a joint loss simultaneously penalizes the degradation of task performance and regularizes the distribution of sampled points, which is consistent with the objective function in Eq. (3). Note that constraints $\mathbf{S} \geq 0$ and $\mathbf{1}_n^{\mathsf{T}}\mathbf{S} = \mathbf{1}_m^{\mathsf{T}}$ in Eq. (3) are naturally guaranteed by the annealing softmax operation for generating the sampling matrix in Section IV-C. For the constraint $\|\mathbf{S}^{\mathsf{T}}\mathbf{S} - \mathbf{I}_m\|_F < \epsilon$, we do not optimize it directly because of the quadratic computation and memory cost. However, our design could implicitly promote $\mathbf{S}^{\mathsf{T}}\mathbf{S}$ to be an identity matrix. We refer readers to the following Figure 4 and Table I for the experimental verification on the learned sampling matrix $\mathbf{S}$.

We show that MOPS-Net is flexible and it can be extended so that a *single* network with *one-time* training is capable of handling *arbitrary* downsampling ratio (Section IV-F). Last but not the least, we prove that MOPS-Net is *permutation-invariant*, which is a highly desired feature for point cloud applications.

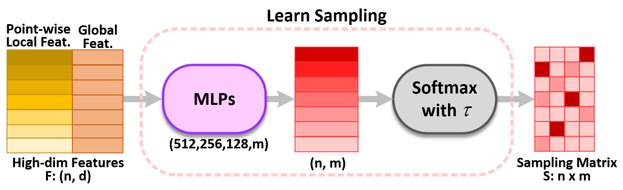


Fig. 3. The flowchart of differentiable sampling module.

Remarks. The proposed MOPS-Net is different from the existing deep learning framework S-Net [29] and SampleNet [30] although they all apply a generative process to obtain downsampled point sets. S-Net and SampleNet formulate the sampling process as a point-generation problem from global features, while MOPS-Net is designed from a matrix optimization perspective to utilize the informative local (or point-wise) features to generate a relaxed sampling matrix. Experimental results demonstrate the advantages of MOPS-Net over S-Net and SampleNet on point cloud classification, reconstruction, and registration. See Section V.

B. Feature Extraction

Given a point cloud $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^n$, we extract d-dimensional point-wise features, denoted by $\mathcal{F} = \{\mathbf{f}_i \in \mathbb{R}^d\}_{i=1}^n$. Let $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_n]^\mathsf{T} \in \mathbb{R}^{n\times d}$ denote the matrix form of pointwise features. We utilize PointNet [13], a basic feature extraction backbone over 3D point clouds. More specifically, as Figure 2 illustrated, we extract pointwise local features via a shared MLP and concatenate them with the global feature which is derived by applying the max-pooling operation.

It is also worth noting that other advanced feature extraction techniques, such as PointNet++ [12], DGCNN [16], and KPConv [70], could be adopted to further boost the performance of our method. , Here we adopt the basic PointNet for fair comparisons with S-Net and SampleNet.

C. Learning Differentiable Sampling Matrix

As analyzed in Section III, an ideal sampling matrix, which is a submatrix of a permutation matrix, is discrete and non-differentiable, making it challenging to optimize. Accordingly, such a non-differentiable sampling matrix cannot be implemented in a deep neural network. Fortunately, the relaxation on the sampling matrix in Eq. (3) leads to a continuous matrix with additional constraints, which approximates the ideal one and allows us to design a deep neural network.

According to Eq. (3), it is known that the optimized sampling matrix $\mathbf{S}$ will depend on $\mathcal{P}$ in a non-linear fashion. From the perspective of geometry processing, downsampling highly depends on the geometric structure of the input data. As the high-dimensional embeddings $\mathbf{f}_i$ produced by the feature extraction module already encode such a structure locally and globally, we use them to predict a preliminary sampling matrix $\overline{\mathbf{S}} \in \mathbb{R}^{n\times m}$ row by row via an MLP $\rho(\cdot)$, i.e.,

$$\bar{\mathbf{s}}_i = \rho(\mathbf{f}_i), \tag{4}$$

where $\bar{\mathbf{s}}_i \in \mathbb{R}^{1\times m}$ is the i-th row of $\overline{\mathbf{S}}$. In our experiments, $\rho(\cdot)$ is realized by a 4-layer MLP of size $[512,\ 256,\ 128,\ \mathrm{m}]$.

TABLE I
Statistic of the percentage of non-zero elements in learned sampling matrix $\mathbf{S}$ ($n = 1024$). r is the threshold.

r	$m = 32$	$m = 64$	$m = 128$
0.01	0.48%	0.36%	0.33%

To satisfy the constraints on the sampling matrix in Eq. (3), we further apply the softmax with temperature annealing operation on each element $\bar{s}_{ij}$ of $\overline{\mathbf{S}}$:

$$s_{ij} = \frac{e^{\bar{s}_{ij}/\tau}}{\sum_{i=1}^n e^{\bar{s}_{ij}/\tau}}, \tag{5}$$

where s_{ij} and is the (i,j)-th entry of the final sampling matrix $\mathbf{S}$, and the value of the temperature τ gradually decreases from 1 to τ_{min} during training, and it is fixed to τ_{min} during inference. Such an operation ensures s_{ij} to be non-negative and encourages each column of $\mathbf{S}$ to be dominated by a single element, especially when τ is small. We experimentally found that such an operation is able to realize the constraints in Eq. (3) well. As visualized in Figure 4, the learned sampling matrix $\mathbf{S}$ is extremely sparse and close to a sample matrix. Besides, $\mathbf{S}^\mathsf{T}\mathbf{S}$ is also close to an identity matrix, although we do not explicitly minimize this constraint.

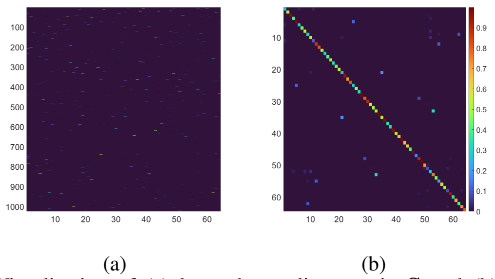


Fig. 4. Visualization of (a) learned sampling matrix $\mathbf{S}$ and (b) $\mathbf{S}^\mathsf{T}\mathbf{S}$ for $m = 64$. The two subfigures share the same color bar.

D. Efficient Regression of the Sampled Set

Having obtained the sampling matrix $\mathbf{S}$, we can naturally deduce the corresponding downsampled point set $\mathbf{Q}$ by

$$\mathbf{Q} = \mathbf{S}^\mathsf{T}\mathbf{P}. \tag{6}$$

Analytically, the downsampling procedure described in Eq. (6) requires explicitly storing the sampling matrix $\mathbf{S} \in \mathbb{R}^{n\times m}$ and performing dot-product with $\mathbf{P}$ in $\mathcal{O}(nm)$ time complexity, which seems to be a major computational bottleneck faced with large-scale point clouds. Fortunately, benefiting from the structural sparsity of $\mathbf{S}$, we can perform the matrix multiplication operation in a much more efficient manner during inference.

More specifically, driven by the temperature annealing operation in Eq. (5), the learned sampling matrix $\mathbf{S}$ highly approximates an ideal sampling matrix, i.e., a column-truncated binary permutation matrix with only m non-zero entries, and thus most entries of $\mathbf{S}$ are *extremely* small (see Figure 4) and make negligible contributions to the actual subset selection. In practice, we design a deterministic matrix simplification rule by setting all entries of $\mathbf{S}$ smaller than an appropriate threshold to zero, after which we can expect that the ratio of non-zero entries should be as small as $\frac{1}{n}$. Particularly, with the threshold set to 0.01 in our implementation, we counted the percentage of non-zero elements in $\mathbf{S}$. As listed in Table I, we can observe that the number of non-zero elements of the quantized $\mathbf{S}$ is about cm, where c is around 4. In our implementation, we adopt the Coordinate format (COO) [71] to store the highly sparse matrix $\mathbf{S}$, which has $\mathcal{O}(cm)$ memory cost, where $c \ll n$. Therefore, performing dot-product on only non-zero elements has time complexity $\mathcal{O}(cm)$.

As relaxation cannot guarantee $\mathcal{Q} \subset \mathcal{P}$, we can adopt an optional post-matching operation that maps each point of $\mathcal{Q}$ to its nearest point in $\mathcal{P}$ to obtain the downsampled subset. If the number of points contained in the matched point set is smaller than the specified value, we adopt FPS to complete it, i.e., points with farthest distances to the matched point set are iteratively selected from the original dense point clouds, until the target number is achieved. In Section V, we will demonstrate the performance of the three cases.

In addition to differentiability, our design of the sampling matrix learning is also permutation-invariant. That is, the sampling result is independent of the point order of the input data. We refer readers to the *Supplementary Material* for the proof.

E. Joint Training Loss

As analyzed in the objective function (3), two types of losses are needed to train MOPS-Net, i.e., the task loss $L_{task}(\cdot)$ and the subset loss $L_{subset}(\cdot,\cdot)$. Specifically, $L_{task}(\cdot)$ aims to promote the network to learn downsampled point clouds that are able to maintain high performance for a specific task. Let $\mathscr{F}_T(\cdot)$ be the network for a typical task, which was trained with the original dense point cloud data, and we have

$$L_{task}(\mathcal{Q}) = L_T(\mathscr{F}_T(\mathcal{Q}), y^*), \tag{7}$$

where y^* is the corresponding ground-truth data for $\mathcal{Q}$. Specifically, y^* will be the class label and the input point cloud when the task is classification and reconstruction, respectively.

The subset loss $L_{subset}(\cdot,\cdot)$ regularizes the network to learn downsampled point clouds that are close to subsets of inputs, which is expressed as

$$L_{subset}(\mathcal{P},\mathcal{Q}) = \frac{1}{m}\sum_{i=1,\ldots,m} \min_{\mathbf{p}\in\mathcal{P}} ||\mathbf{q}_i - \mathbf{p}||_2^2 \tag{8}$$

Therefore, the total loss $L_{total}(\cdot,\cdot)$ for end-to-end training of MOPS-Net is written as

$$L_{total}(\mathcal{P},\mathcal{Q}) = L_{task}(\mathcal{Q}) + \alpha L_{subset}(\mathcal{P},\mathcal{Q}), \tag{9}$$

where $\alpha > 0$ balances the two terms. Figure 16 shows the effect of the value of α on performance.

F. Flexible MOPS-Net for Arbitrary Ratios

In the previous sections, we construct MOPS-Net with a predefined sample size m, and a different network has to be trained for each m, which is tedious and unpractical for real-world applications. To solve this issue, we extend MOPS-Net and propose flexible MOPS-Net (FMOPS-Net), which is a single network that can achieve 3D point cloud downsampling with arbitrary sampling ratios after only one-time training.

Specifically, we consider learning a relatively large matrix $\widehat{\mathbf{S}} \in \mathbb{R}^{n\times m_{max}}$ with the same network architecture as MOPS-Net. Given an arbitrary sample size $m \le m_{max}$, we select the m left-most columns of $\widehat{\mathbf{S}}$ to form the sampling matrix $\mathbf{S}_m \in \mathbb{R}^{n\times m}$, producing a point cloud $\mathcal{Q}_m$ with m points according to Eq. (6). Such a manner is equivalent to indirectly sorting the points of $\mathcal{P}$ according to their importance in a downsampled point cloud. To enable flexibility, we train MOPS-Net by randomly picking the downsampled number $m \le m_{max}$ and minimizing $L_{total}(\mathcal{P},\mathcal{Q}_m)$ at each iteration. Note that the computational complexity and the memory consumption during the inference phase are identical to Section IV-D by replacing m with m_{max}.

V. Experiments

We validated the effectiveness of MOPS-Net and FMOPS-Net on three typical tasks, i.e., point cloud classification, reconstruction, and registration. The task networks and experiment settings will be discussed in each section in detail. We used three widely used traditional downsampling methods, i.e., random sampling (RS), voxelization (Voxel), and farthest point sampling (FPS), as baselines. We also compared with S-Net [29] and SampleNet [30], which are state-of-the-art deep learning-based task-oriented downsampling methods. Note that for S-Net, SampleNet, and MOPS-Net, a network was trained for a downsampling size. For deep learning-based methods, we examined the performance of three types of downsampled sets, i.e., (1) **Generated (G)** sets: the point sets are generated directly by deep learning-based methods; (2) **Matched (M)** sets: the directly generated sets are post-processed via the matching operation, making the point sets be subsets of input ones; and (3) **Completed (C)** sets: the matched sets are further completed via FPS if their numbers of points are less than the specified value. In the following visual results, we visualized the **Generated (G)**, **Matched (M)**, and **Completed (C)** sets with blue, red, and orange colors, respectively, to distinguish them.

A. Classification-oriented Downsampling

1) Classification of small-scale point clouds: Following S-Net [29] and SampleNet [30], we used the pre-trained PointNet vanilla [13] performing on ModelNet40 [72] as the classification task network. Note that the pre-trained PointNet vanilla trained on point clouds with 1024 points achieves 87.1% overall accuracy when classifying point clouds with 1024 points each to 40 categories. The task loss refers to the cross entropy between the predicted and ground-truth labels. During the training of our MOPS-Net, we optimized the network with the Adam optimizer and set $\tau_{min} = 0.1$ and $\alpha = 30$, and we initialized the learning rate to $5e^{-4}$ and exponentially decreased it to $1e^{-5}$ within 250 epochs.

Quantitative comparisons. From Table II, we can observe that task-oriented downsampling methods, including S-Net, SampleNet, and MOPS-Net, achieve much better performance

TABLE II
COMPARISONS OF THE CLASSIFICATION ACCURACY[2] BY DIFFERENT DOWNSAMPLING METHODS. THE LARGER, THE BETTER.

	RS	Voxel	FPS	S-Net [29]			SampleNet [30]			MOPS-Net			FMOPS-Net		
m				G	M	C	G	M	C	G	M	C	G	M	C
512	84.76	73.82	86.06	81.69	71.43	85.66	57.82	57.82	86.63	86.67	85.25	**86.75**	86.18	85.07	86.35
256	76.94	73.50	83.06	82.94	72.08	82.78	83.23	83.02	84.48	**86.63**	85.53	86.10	86.51	85.17	86.14
128	62.32	68.15	72.85	83.31	72.24	73.18	84.04	83.14	82.58	86.06	**84.64**	85.29	86.02	84.4	84.89
64	36.75	58.31	56.69	78.81	66.00	63.82	82.21	80.19	79.78	**85.25**	83.95	84.00	83.39	81.44	81.08
32	15.07	20.02	34.08	78.16	59.89	60.05	75.45	72.89	72.49	**84.28**	79.01	79.74	76.58	71.76	70.42
16	6.36	13.94	20.22	68.56	43.60	43.44	54.42	50.04	50.00	**81.40**	64.99	64.83	67.22	56.77	55.71
8	4.70	3.85	11.02	45.99	20.50	20.5	29.58	25.53	25.57	**52.39**	32.62	32.62	40.96	32.62	32.58

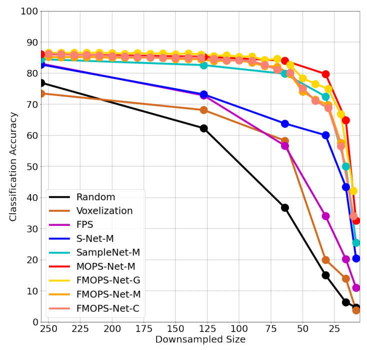


Fig. 5. Quantitative comparisons of classification by different downsampling methods. Note that FMOPS-Net is applicable for arbitrary downsampled size.

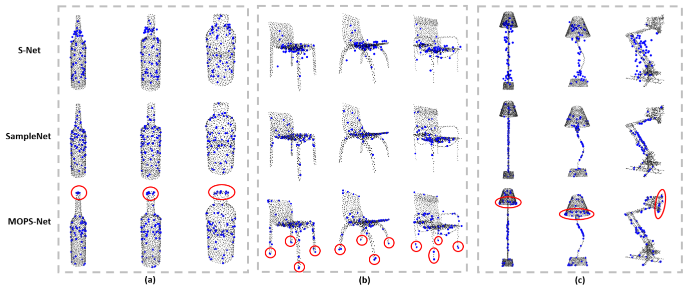


Fig. 6. Visual comparisons of the generated point sets by the three deep learning-based task-oriented downsampling methods with $m = 64$ over three classes: (a) Bottles (b) Chairs (c) Lamps. Prominent regions within each class are highlighted in red.

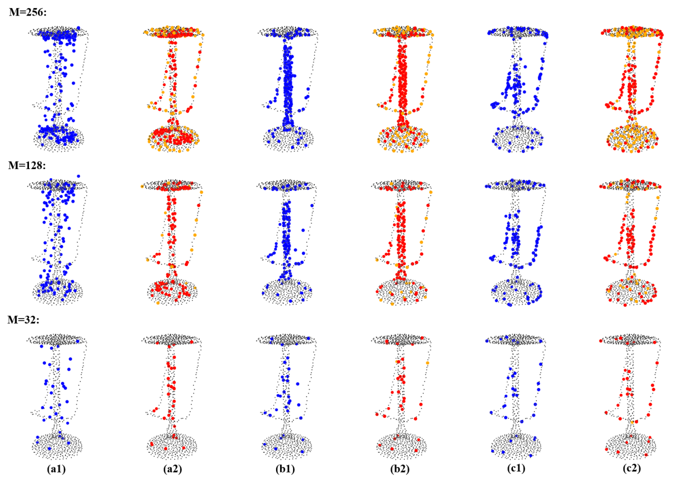


Fig. 7. Visualization of sampled point clouds by different methods for m = 256, 128, and 32. (a1) generated sets by S-Net; (a2) matched and completed sets by S-Net; (b1) generated sets by SampleNet; (b2) matched and completed sets by SampleNet; (c1) generated sets by MOPS-Net; (c2) matched and completed sets by MOPS-Net.

than task-independent methods, including random sampling (RS), voxelization, and FPS. Note that the accuracy of S-Net drops significantly from generated sets to the matched subsets because the generative-based S-Net fails to obey the subset constraint. By replacing the repeated points in matched subsets by FPS points, the accuracy of completed subsets by S-Net can be improved, especially for relatively large downsampled sizes m =256 and 512.

SampleNet improves S-Net by projecting the generated sets to nearest neighbors in original point clouds, and thus can minimize the performance gap between the generated set and matched subset. However, because of the additional restriction for projection, the accuracy of generated points by SampleNet is inferior to that by S-Net for relatively small downsampled sizes $m = 8, 16, 32$. Moreover, SampleNet fails to generate meaningful points for $m = 512$, and it mainly relies on additional FPS postprocessing to obtain the completed set which can result in comparable performance.

The proposed MOPS-Net consistently achieves the best performance over all cases, which is credited to the real sampling process-like modeling of MOPS-Net. Note that FMOPS-Net is a flexible model that only requires one-time training to handle arbitrary downsample ratios. Compared with MOPS-Net, the FMOPS-Net variant additionally learns to sort input points on the basis of relative importance in the downsampled point cloud. Thus, FMOPS-Net is intrinsically faced with a greater learning difficulty and suffers from slight degradation of the accuracy of MOPS-Net. However, FMOPS-Net still outperforms S-Net and SampleNet in almost all cases. Figure 5 further demonstrates the flexibility and advantage of FMOPS-Net by showing the accuracy of more downsampling sizes.

We also emphasize that performance under larger downsampling ratios is a more persuasive measurement for point cloud downsampling algorithms. Typically, for smaller downsampling ratios (i.e., larger m), the resulting point clouds are still dense enough to involve sufficient information for downstream processing. Under such circumstances, the impact of the applied downsampling algorithm is weakened. By contrast, under larger downsampling ratios (i.e., smaller m) where only a small number of points can be preserved, the impact of specific downsampling strategy is strengthened. In other words, under a much more limited selection budget, the downsampling model must select the most informative points more carefully, which turns out to be a more demanding evaluation protocol. From this perspective, this phenomenon is a stronger indicator of the superiority of MOPS-Net compared with other competing methods.

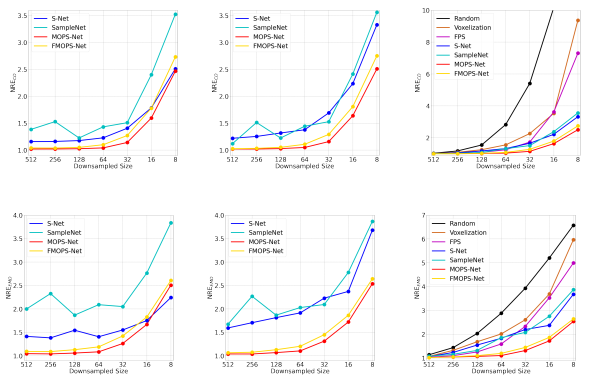
Visual comparisons. Figure 6 visually illustrates sampled point clouds ($m = 64$) by the three deep learning-based task-oriented downsampling methods, i.e. S-Net, SampleNet, and MOPS-Net, on three classes. From Figure 6, it can be seen that the two generative-based S-Net and SampleNet tend to generate points close to the centers of shapes and fail to capture prominent regions. By contrast, our MOPS-Net can successfully select points near the contours of shapes and focus on prominent regions. Besides, for different point clouds of a typical class, MOPS-Net focuses on selecting points corresponding to identical semantics, e.g. the bottleneck of bottles, the legs of chairs, and the lampshade of lamps. These observations also explain why the downsampled point clouds by MOPS-Net can be classified with higher accuracy than S-Net and SampleNet.

Besides, Figure 7 visualizes the generated, matched, and completed sets by S-Net, SampleNet, and MOPS-Net under various m, where it can be seen that S-Net fails to directly generate downsampled points close to the input points once the shape is complex, and SampleNet tends to directly generate points clustered around the shape center and omit the footrest of the stool, which is assumed to be important for shape identification. By contrast, our MOPS-Net can consistently capture these discriminative regions for any downsampled size.

2) Classification of large-scale point clouds: To demonstrate the ability of MOPS-Net in processing large-scale point clouds, we further applied MOPS-Net for downsampling point clouds with 10,000 and 100,000 points each. We first pre-trained a PointNet-vanilla classifier on ModelNet40 with 100,000 points each model. The other settings, including the optimized loss function, training epoch, and training strategies, were identical to the experiments in Section V-A1. The pre-trained classifier can achieve 90.11% overall accuracy. By fixing the pre-trained classifier, we then trained MOPS-Net for downsampling point clouds with 10,000 points each. The experiment settings were kept identical to our previous experiments on small-scale point clouds. As MOPS-Net is built upon point-wise MLPs and the Softmax operator, which are independent of the number of input points, we can directly apply MOPS-Net trained on 10,000 points for downsampling larger-scale point clouds with 100,000 points each.

Table III lists the classification accuracy of matched sets by MOPS-Net for downsampling 10,000 points and 100,000 points. The high classification accuracy demonstrates the ability of the proposed MOPS-Net on downsampling large-scale point clouds. In particular, MOPS-Net trained on point clouds with 10,000 points each can be successfully extended to downsample larger-scale point clouds with 100,000 points each, without any modification or fine-tuning, which also demonstrates its flexibility.

[2]Note that in the original papers, S-Net and SampleNet adopt the pre-trained PointNet with the T-Net structure whose classification accuracy achieves 89.2%. As aforementioned, in this paper we utilize PointNet *without* the T-Net structure as the classifier to enable the application on large-scale point cloud data. For a fair comparison, we apply this classifier to all compared methods. Thus, the results of S-Net and SampleNet in this paper are slightly different from those of the original papers [29] [30].



(a) (b) (c)

Fig. 8. Quantitative comparisons of the distortion of reconstructed dense point clouds from the corresponding downsampled ones by different downsampling methods. Reconstructed CD (first row) and EMD (second row) from (a) generated, (b) matched, and (c) completed sets.

B. Reconstruction-oriented Downsampling

1) Reconstruction of small-scale point clouds: In this scenario, we followed the settings of S-Net and SampleNet to evaluate our method. The task network $\mathscr{F}_T(\cdot)$ was achieved by a pre-trained MLP-based reconstruction network[50], where a 3-layer MLP of size $[256,\ 256,\ 1024 \times 3]$ is utilized to reconstruct point clouds with 1024 points from a 128-dimensional global feature. We obtained the 128-dimensional global features of downsampled point clouds by applying max-pooling on the pointwise features extracted by a 5-layer MLP of size $[64,\ 128,\ 128,\ 256,\ 128]$. A single class of ShapeNetCore [73] was used for training and testing. The task loss was set as the combination of the Chamfer distance (CD) and earth-mover distance (EMD). During training of our MOPS-Net, we set $\tau_{min} = 0.5, \alpha = 0.2$ and initialized the learning rate to $5e^{-4}$ and exponentially decreased it to $1e^{-5}$ within 250 epochs. We quantitatively measured the reconstruction performance of different downsampling methods using the normalized reconstruction error (NRE) for CD and EMD, which are defined as

$$\mathsf{NRE}_{\mathsf{CD}}(\mathcal{Q}, \mathcal{P}) = \frac{\mathsf{CD}(\mathcal{P}, \mathscr{F}_T(\mathcal{Q}))}{\mathsf{CD}(\mathcal{P}, \mathscr{F}_T(\mathcal{P}))}. \tag{10}$$

$$\mathsf{NRE}_{\mathsf{EMD}}(\mathcal{Q}, \mathcal{P}) = \frac{\mathsf{EMD}(\mathcal{P}, \mathscr{F}_T(\mathcal{Q}))}{\mathsf{EMD}(\mathcal{P}, \mathscr{F}_T(\mathcal{P}))}, \tag{11}$$

where $\mathsf{CD}(\cdot)$ and $\mathsf{EMD}(\cdot)$ compute the CD and EMD, respectively. The values of $\mathsf{NRE}_{\mathsf{CD}}$ and $\mathsf{NRE}_{\mathsf{EMD}}$ are lower bounded by 1, and the smaller, the better.

Quantitative comparisons. Figures 8(a) and (b) show the $\mathsf{NRE}_{\mathsf{CD}}$ and $\mathsf{NRE}_{\mathsf{EMD}}$ values of the reconstructed point clouds from generated and matched sets by different donwsampling methods under various downsampling sizes, where it can be seen that except the extremely small downsampled size ($m = 8$ and 16), our MOPS-Net and FMOPS-Net produce

TABLE III
COMPARISONS OF CLASSIFICATION ACCURACY FOR DOWNSAMPLING LARGE-SCALE POINT CLOUDS. THE LARGER, THE BETTER.

	$n = 10,000$					$n = 100,000$				
m	RS	FPS	S-Net [29]	SampleNet [30]	MOPS-Net	RS	FPS	S-Net [29]	SampleNet [30]	MOPS-Net
1024	84.97	88.65	73.18	81.89	**90.24**	84.93	88.98	72.61	81.89	**89.79**
512	77.23	85.69	72.45	78.69	**90.00**	76.86	85.86	72.12	78.32	**89.18**
256	65.84	79.67	63.09	84.68	**87.40**	64.83	79.78	63.41	84.85	**89.10**

TABLE IV
QUANTITATIVE COMPARISONS OF DIFFERENT RECONSTRUCTION FRAMEWORKS. THE SMALLER, THE BETTER. MODEL SIZES WERE MEASURED UNDER THE SAME PARAMETER PRECISION.

				M-FoldingNet					
points	metric	MLP-based	FoldingNet [74]	$M = 4$	$M = 8$	$M = 16$	$M = 32$	$M = 64$	$M = 128$
1024	CD	13.81	18.48	**11.84**	12.69	13.46	13.34	13.70	16.46
	EMD	**154.26**	359.78	175.07	209.13	210.69	210.69	198.66	222.89
	Model Size	4.1MB	4.7 MB	3.1MB	2.9 MB	2.8MB	2.8MB	2.8MB	2.8MB
10000	CD	NA	1.25	0.84	0.58	0.49	0.42	**0.36**	0.36
	EMD	NA	18.8	18.3	11.27	7.31	4.86	**2.56**	2.70
	Model Size	NA	12.0MB	5.4 MB	4.4MB	3.9 MB	3.6MB	3.5MB	3.5MB

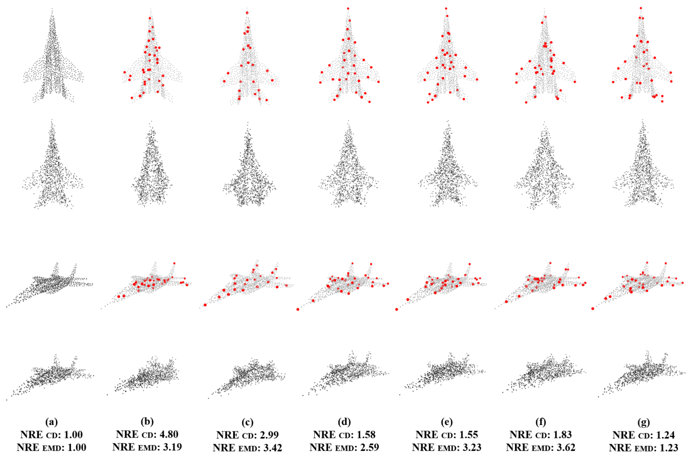


Fig. 9. Visual comparisons of downsampled point clouds (1st and 3rd rows) and reconstructed point clouds (2nd and 4th rows) by different downsampling methods with $m = 32$. (a) Original point clouds and corresponding reconstructions. (b) Random sampling; (c) Voxelization; (d) FPS; (e) S-Net; (f) SampleNet; (g) MOPS-Net.

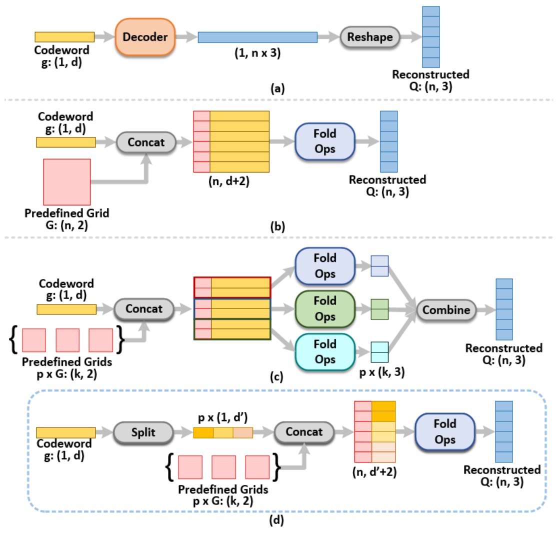


Fig. 10. Illustration of the differences between different frameworks for point cloud reconstruction. (a) MLP-based; (b) FoldingNet [74]; (c) AtlasNet [75]; (d) Proposed M-FoldingNet. Note that the multiple folding operators of the AtlasNet are independent. For FoldingNet, AtlasNet, and M-FoldingNet, the number of reconstructed points depends on the dimensions of the 2D grids.

much lower distortion than S-Net and SampleNet. For the reconstruction from completed sets shown in Figure 8(c), S-Net and SampleNet are even worse than FPS when $m > 64$. The reason may be that FPS is able to produce uniformly distributed downsampled points; however, the uniform distribution cannot be guaranteed by S-Net and SampleNet as they are generative methods. However, our MOPS-Net and FMOP-Net consistently achieve the best and second-best performance under all downsampled sizes, respectively, and FMOP-Net is even comparable to MOPS-Net.

Visual comparisons. Figure 9 visually compares the reconstructed dense point clouds from sparse ones obtained by different downsampling methods, where it can be seen that the reconstructed point clouds from our MOPS-Net are much better than those from other downsampling methods and are much closer to ground-truth ones. This advantage is credited to that the downsampled points by our MOPS-Net can well capture the contour and salient features of 3D shapes.

2) Reconstruction of large-scale point clouds: To demonstrate the ability of our MOPS-Net on large-scale point clouds, we also examined MOPS-Net on point clouds with 40,960 points each. Unfortunately, the MLP-based reconstruction framework (see Figure 10(a)) employed in Section V-B1 cannot well adapt to large-scale point cloud reconstruction because the network size is linearly proportional to the number of output points. To this end, we also propose a new framework for reconstructing large-scale point clouds, whose network size is independent of the output point number.

The proposed framework for large-scale point cloud reconstruction, namely Multi-FoldingNet (M-FoldingNet), is motivated by FoldingNet [74] shown in Figure 10(b). As illustrated in Figure 10(d), instead of concatenating the global feature to the 2D coordinates of a single regular grid, we first segment such a global feature into a set of M local features with a

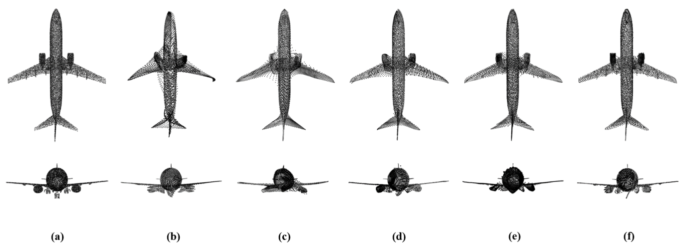


Fig. 11. Visual comparisons of the reconstructed large-scale point clouds via different reconstruction frameworks. (a) Original point cloud with 10, 000 points; Reconstructed by (b) FoldingNet; (c) M-FoldingNet ($M = 4$); (d) M-FoldingNet ($M = 8$); (e) M-FoldingNet ($M = 16$); (f) M-FoldingNet ($M = 64$).

smaller and equal feature dimension, and it is expected that each local feature encodes the high-level semantic information of a local patch on a point cloud. We then concatenate each local feature to the coordinates of a 2D regular grid separately, which are fed into a shared folding operator. Note that the number of output points can be varied by adjusting the dimensions of the 2D regular grids. Compared with FoldingNet, which can be thought of as a special case of M-FoldingNet with $M = 1$, M-FoldingNet with fewer network parameters can allow the folding operator to focus on local regions which are easier to be reconstructed. Although AtlasNet [75] illustrated in Figure 10(c) also realizes reconstruction in a local manner, it adopts multiple independent folding operators, leading to the significant increase of network parameters, compared with FoldingNet.

We first evaluated and compared the proposed M-FoldingNet with other reconstruction frameworks on both small-scale point clouds (1024 points) and large-scale point clouds (10,000 points each)[3]. The settings, including the dataset, the generation of codewords/global features, and the task loss, were kept identical to the MLP-based reconstruction framework in Section V-B1. For fair comparisons, we used an identical codeword dimension for all reconstruction frameworks, i.e., the codeword dimension equals to $d = 128$ (resp. $d = 512$) for reconstructing point clouds with $n = 1024$ (resp. $n = 10,000$) points. As listed in Table IV, we can observe that the proposed M-FoldingNet can achieve better performance than FoldingNet. Specifically, M-FoldingNet with $M = 4$ can achieve the best performance on the reconstruction of small-scale point clouds. The reconstruction quality decreases with M increasing because the segmented local features have a limited dimension to fully embed local part information. As expected, more pieces of local features are needed to achieve the best reconstruction performance. Besides, our M-FoldingNet is more compact than FoldingNet and MLP-based. Figure 11 visually compares the reconstructed point clouds by M-FlodingNet and FoldingNet, which also demonstrates the superiority of the proposed reconstruction framework

We evaluated the performance of MOPS-Net for downsampling real scanned data with 40,960 points each [76], where the pre-trained M-FoldingNet ($M = 128$ and $d = 2048$ (or $d' = 16$)) was used as the task network. The settings of MOPS-Net were the same as those in Section V-B1. Table V quantitatively compares the distortion of reconstructed point clouds from the downsampled sparse point clouds by random sampling, FPS, and MOPS-Net [4]. From Table V, we can see that the $\mathrm{NRE_{CD}}$ and $\mathrm{NRE_{EMD}}$ values of the reconstruction from the downsampled points by MOPS-Net are much smaller than those of the reconstruction from downsampled points by RS and FPS under all cases, which demonstrates the ability of MOPS-Net in handling large-scale point clouds. Besides, in Figure 12, we visualized the reconstructed dense point clouds from downsampled $m = 256$ points via different methods. The high quality of reconstructed 40,960 dense point clouds from MOPS-Net demonstrates the superiority of the proposed method for downsampling large-scale point clouds. The memory and computational complexity for downsampling large-scale point clouds can also be found in Section V-F. Last but not least, for a very large-scale point cloud in practice, a promising solution is to partition it to several regions with smaller points each and then downsample the regions in parallel or sequentially.

TABLE V
QUANTITATIVE COMPARISONS OF THE DISTORTION OF RECONSTRUCTED POINT CLOUDS FROM DOWNSAMPLED SPARSE POINT CLOUDS BY DIFFERENT METHODS. THE ORIGINAL POINT CLOUDS CONSIST OF 40,960 POINTS EACH. THE SMALLER, THE BETTER.

	$\mathrm{NRE_{CD}}$			$\mathrm{NRE_{EMD}}$		
m	RS	FPS	MOPS	RS	FPS	MOPS
4096	1.37	1.33	**1.06**	3.93	3.23	**1.57**
1024	4.00	2.53	**1.35**	12.10	8.97	**3.05**
256	26.77	7.24	**2.71**	45.53	19.13	**7.40**

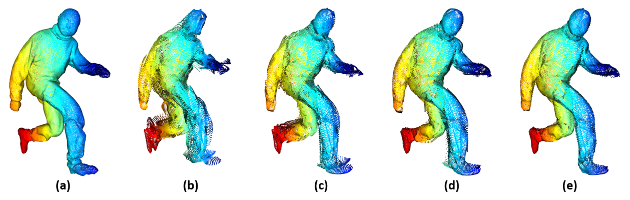


Fig. 12. Visual comparisons of reconstructed large-scale point clouds by different downsampling methods with $m = 256$. (a) Original 40,960 real scanned data. Reconstructed dense point clouds from 256 points sampled by (b) Random sampling; (c) FPS; (d) MOPS-Net; (e) Reconstructed dense point cloud by extracting the codeword from the original point cloud. Colors are assigned by the pointwise depths for better visualization.

C. Registration-oriented Downsampling

Registration aims to predict rigid transformations between two point clouds, including a rotation and a translation, which can well align them. As an overdetermined problem, only a few key points, which can well capture the shape information of a point cloud, are usually extracted, and the registration

[3]Here we used point clouds with 10,000 points each to enable the application of the MLP-based framework.

[4]Note that we did not provide the results of S-Net and SampleNet in this experiment because it is difficult to tune the hyper-parameters contained in the loss functions of S-Net and SampleNet for obtaining satisfied results due to the large-scale point clouds. To ensure the correctness of our paper, we omitted their results. Besides, in Section V-E, we quantitatively analyzed the effect of the hyper-parameters contained in the loss function of SampleNet on reconstruction performance.

will be conducted on the key points rather than original point clouds to save memory and computational complexity. Thus, the registration accuracy also depends on the quality of selected key points. In this section, we examined the performance of MOPS-Net with registration as the subsequent task.

We utilized point clouds with 1024 points each in ModelNet40 as the original data. The paired data were generated by applying random rotations and translations to training point clouds. We adopted PCRNet [23] with one iteration as the task network, whose loss function is the L2 difference between the predicted quaternions and ground-truth ones. We quantitatively evaluated different downsampling methods by using the mean rotation error (MRE) between the predicted rotations and ground-truth ones. Note that the MRE of PCRNet trained and tested with original point clouds is 7.21. For MOPS-Net, we set $\tau_{min} = 0.1, \alpha = 1$ and initialized the learning rate to $1e^{-4}$ and exponentially decreased to $1e^{-5}$ within 250 epochs.

Quantitative and visual comparisons. Table VI lists MRE values of different downsampling methods with the task network fixed to the pre-trained PCRNet, where we can observe that the proposed MOPS-Net can achieve the best performance than the traditional methods, S-Net, and SampleNet for all settings, and FMOPS-Net even achieves better performance than MOPS-Net over the generated sets. Besides, S-Net suffers from significant performance degradation once the generated point sets are restricted to be subsets of original point clouds (i.e. the matched sets). Figure 13 visually compares different methods, where the advantage of our MOPS-Net is verified again.

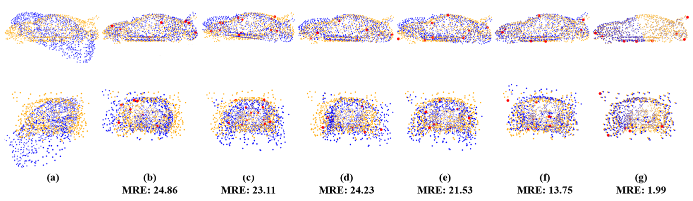


Fig. 13. Visual comparisons of registered point clouds by different downsampling methods ($m = 8$). (a) Non-registered input pair; Registered results on the key points extracted by (b) Random sampling; (c) Voxelization; (d) FPS; (e) S-Net; (f) SampleNet; and (g) MOPS-Net.

Joint training. In all the above experiments for classification, reconstruction, and registration, the task networks were fixed to be the pre-trained models. Here, taking the registration task as an example, we illustrated the advantage of joint training i.e., the task network PCRNet and the downsampling network MOPS-Net are jointly trained. As listed in Table VII, such a joint training manner can further improve registration accuracy.

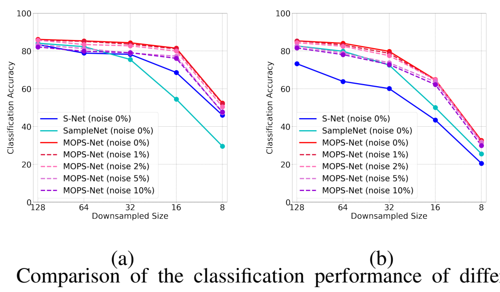


Fig. 14. Comparison of the classification performance of different downsampling methods applied to point clouds with various levels of noise. (a) performance on the generated sets (b) performance on the completed sets Note that the noise level refers to that added to each dimensional of 3D point cloud data.

TABLE VII
QUANTITATIVE COMPARISONS OF MREs FOR PRE-TRAINED AND JOINTLY TRAINED REGISTRATION NETWORKS. THE SMALLER, THE BETTER.

	Pre-trained PCRNet			Jointly trained PCRNet		
m	G	M	C	G	M	C
64	8.58	8.56	8.00	5.96	8.18	8.22
32	7.97	8.53	8.54	6.74	9.18	9.65
8	12.63	12.68	12.68	6.94	11.15	11.15

D. Robustness Analysis

We also evaluated the robustness of the proposed MOPS-Net to noise over the classification task. We added various levels of Gaussian noise to input point clouds. As shown in Figure 14, even the input point clouds are highly contaminated, i.e., the noise level in each dimension is 10%, MOPS-Net still remains high accuracy which is comparable to that of MOPS-Net with clean input, demonstrating its robustness. Besides, the accuracy of MOPS-Net with noisy input is still higher than that of S-Net and SampleNet with clean input.

In Figure 15, we visually illustrated the downsampled points by our MOPS-Net over noisy data, where it can be seen that the proposed MOPS-Net can capture the important regions (head, hands, legs) and the locations of sampled points remain consistent at different noise levels.

E. Ablation Study

Taking the reconstruction task in Section V-B1 as an example, we investigated the effect of the hyper-parameters contained in the loss functions of our MOPS-Net and SampleNet [30]. The loss function used by SampleNet is given as

$$L_{SampleNet} = L_{task} + \beta L_{projection} + \gamma L_{simplify}, \tag{12}$$

TABLE VI
QUANTITATIVE COMPARISONS OF MREs OF DONWSAMPLED POINT CLOUDS BY DIFFERENT METHODS USED FOR REGISTRATION. THE SMALLER, THE BETTER.

	RS	Voxel	FPS	S-Net [29]			SampleNet [30]			MOPS-Net			FMOPS-Net		
m				G	M	C	G	M	C	G	M	C	G	M	C
64	17.47	10.37	9.98	10.93	12.83	13.26	8.30	8.69	8.29	8.58	8.56	8.00	7.94	8.87	8.84
32	26.95	14.46	13.12	10.23	15.45	14.99	8.48	9.26	9.18	7.97	8.53	8.54	7.78	8.86	8.92
8	61.14	44.80	33.28	13.47	17.67	17.67	13.4	14.88	14.86	12.64	12.68	12.68	12.13	12.98	12.98

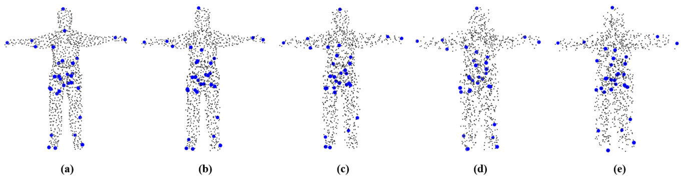
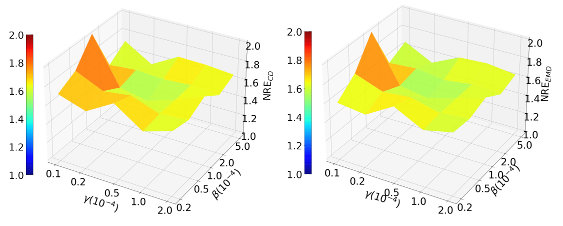
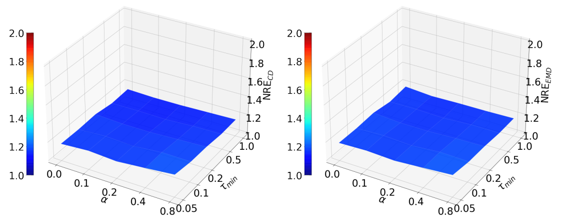


Fig. 15. Visualization of sampled point clouds by MOPS-Net on clean data and noisy data with various levels of noise. (a) clean; (b) 1% noise ; (c) 2% noise; (d) 5% noise; (e) 10% noise. Note that the noise level refers to that added to each dimensional of 3D point clouds.



(a) SampleNet



(b) MOPS-Net

Fig. 16. Illustration of the effect of the hyper-parameters involved in the loss functions of our MOPS-Net and SampleNet on reconstruction performance ($m = 32$). SampleNet's performance is highly sensitive to the hyper-parameters, whereas ours is not.

which contains two hyper-parameters β and γ. Note that in the experiments of Section V-B1, the hyper-parameters of both MOPS-Net and SampleNet have been tuned to be almost optimal, i.e., $\tau_{min} = 0.5$ and $\alpha = 0.2$ for MOPS-Net and $\beta = 1e^{-4}$ and $\gamma = 5e^{-5}$ for SampleNet.

As shown in Figure 16, we can see that our MOPS-Net can consistently achieve almost optimal performance in wide ranges of α and τ_{min}, demonstrating its stability. However, the performance of SampleNet varies largely as the values of β and γ change. Besides, under the best parameter settings, SampleNet is still worse than MOPS-Net in terms of both NRE_{CD} and NRE_{EMD}.

F. Complexity Analysis

Table VIII reports the running time of various methods applied to downsample point clouds with 1024 points each. All methods were implemented on GTX 2080Ti GPU, and we reported the average inference time per shape. From Table VIII, we observe that the running time of FPS is linearly proportional to the downsampled size m, while the other methods are insensitive to m. Besides, S-Net and MOPS-Net are even faster than random sampling. Compared with S-Net, SampleNet requires an additional projection operation involving k-NN search, and thus takes more time than S-Net.

TABLE VIII
AVERAGE RUNNING TIME ($\times 10^{-4}$ SECONDS) FOR DOWNSAMPLING A 1024-POINT MODEL.

m	RS	FPS	S-Net [29]	SampleNet [30]	MOPS-Net
512	1.22	68.28	0.78	3.39	1.00
256	1.26	29.02	0.77	2.95	0.95
128	1.22	15.32	0.75	2.72	1.06
64	1.27	8.68	0.79	2.50	0.99
32	1.32	5.12	0.79	2.39	0.89
16	1.12	2.47	0.72	2.38	0.87
8	1.16	1.46	0.73	2.33	0.98

TABLE IX
TIME EFFICIENCY ($\times 10^{-1}$ SECONDS) FOR DOWNSAMPLING 40,960 POINTS.

m	RS	FPS	S-Net [29]	SampleNet [30]	MOPS-Net
4096	0.07	182.77	8.39	10.07	7.24
1024	0.07	65.46	5.26	8.01	7.04
256	0.07	16.31	5.49	9.08	7.40

TABLE X
MEMORY CONSUMPTION AND RUNNING TIME OF MOPS-NET TO DOWNSAMPLE $m = 4096$ POINTS FROM LARGE-SCALE POINT CLOUDS WITH $n = 40,960$ POINTS.

Module	GPU Memory	Time ($\times 10^{-1}$)
Feature extraction	2123MB	6.59
Learn sampling	2560MB	0.04
Regress points	0MB	0.14
Task network	4MB	0.25

Besides, we also analyzed the complexity of downsampling large-scale point clouds. The running time and memory consumption were recorded during the inference period on Quadro RTX 8000 GPU. Table IX lists the running time of our MOPS-Net applied to downsample point clouds with 40,960 points each. We also provided the running time of random sampling, FPS, and deep learning-based S-Net and SampleNet as a reference. Note that FPS is an iterative searching process that naturally cannot be parallelized by modern GPU devices. Table X lists the memory consumption and running time for each step when downsampling 4096 points from 40,960 points, where it can be observed that performing the standard feature extraction via PointNet on large-scale point clouds consumes most of the inference time. Since MOPS- Net, S-Net and SampleNet all adopt PointNet as the feature extractor, they yield similar efficiency. In the meantime, the proposed sampling modules, including the learning of the sampling matrix and regression of the sampled set are very efficient. Besides, it is worth noticing that the MLP operators, which formulate the feature extraction module and learning of the sampling matrix module, require large memory consumption when dealing with large-scale point clouds.

VI. CONCLUSION & FUTURE WORK

In this paper, we presented MOPS-Net, a novel end-to-end deep learning framework to compactly represent 3D point clouds in a task-oriented manner. In contrast to the existing methods, we designed MOPS-Net from the perspective of matrix optimization. As the original discrete and combinatorial optimization problem is difficult to solve, we obtained a continuous and differentiable form by relaxing the 0-1 constraint of each variable. MOPS-Net elegantly mimics the function of the resulting matrix optimization problem by exploring both local and global structures of input data. MOPS-Net is permutation invariant and can be end-to-end trained with a task network. We applied MOPS-Net to three typical applications, including 3D point cloud classification, reconstruction, and registration, and observed that MOPS-Net produces better results than state-of-the-art methods. Moreover, MOPS-Net is flexible in that with a simple modification, a single network with one-time training can handle arbitrary downsampling ratios. We justified our optimization-driven design principle and demonstrated the efficacy of MOPS-Net through extensive evaluations and comparisons.

The promising results of MOPS-Net inspire several interesting future directions. For example, it can replace the widely used FPS in feature extraction of current networks to boost performance. Though MOPS-Net is designed for point cloud downsampling, increasing the dimension of the differential sampling matrix allows us to handle upsampling as well. Moreover, MOPS-Net opens the door to applying matrix optimization in deep learning. We believe the matrix optimization idea is general and can work for other selection and ranking problems, such as keyframe selection in videos [77], [78], band selection in hyperspectral images [79], [80] and view selection in light field images [81].

References

[1] R. Schnabel and R. Klein, "Octree-based point-cloud compression." in *PBG@ SIGGRAPH*, 2006, pp. 111–120.

[2] V. Morell, S. Orts, M. Cazorla, and J. Garcia-Rodriguez, "Geometric 3d point cloud compression," *Pattern Recognition Letters*, vol. 50, pp. 55–62, 2014.

[3] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1–4.

[4] K. Mammou, P. A. Chou, D. Flynn, M. Krivokuća, O. Nakagami, and T. Sugio, "G-pcc codec description v2," *ISO/IEC JTC1/SC29/WG11 N18189*, 2019.

[5] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 778–785.

[6] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuća, S. Lasserre, Z. Li *et al.*, "Emerging mpeg standards for point cloud compression," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 133–148, 2018.

[7] M. Quach, G. Valenzise, and F. Dufaux, "Learning convolutional transforms for lossy point cloud geometry compression," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 4320–4324.

[8] A. F. Guarda, N. M. Rodrigues, and F. Pereira, "Deep learning-based point cloud coding: A behavior and performance study," in *2019 8th European Workshop on Visual Information Processing (EUVIP)*. IEEE, 2019, pp. 34–39.

[9] T. Huang and Y. Liu, "3d point cloud geometry compression on deep learning," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 890–898.

[10] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, "The farthest point strategy for progressive image sampling," *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1305–1315, 1997.

[11] X. Ying, S. Xin, Q. Sun, and Y. He, "An intrinsic algorithm for parallel poisson disk sampling on arbitrary surfaces," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 9, pp. 1425–1437, 2013.

[12] C. R. Qi, L. Yi, H. Su, and 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.

[13] C. R. Qi, H. Su, K. Mo, and 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.

[14] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *Advances in neural information processing systems*, 2018, pp. 820–830.

[15] Y. Shen, C. Feng, Y. Yang, and D. Tian, "Mining point cloud local structures by kernel correlation and graph pooling," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4548–4557.

[16] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, pp. 1–12, 2019.

[17] J. Li, B. M. Chen, and G. Hee Lee, "So-net: Self-organizing network for point cloud analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9397–9406.

[18] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," in *2017 international conference on 3D vision (3DV)*. IEEE, 2017, pp. 537–547.

[19] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and 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.

[20] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and 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. 11 108–11 117.

[21] D. Li, G. Shi, Y. Wu, Y. Yang, and M. Zhao, "Multi-scale neighborhood feature extraction and aggregation for point cloud segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 6, pp. 2175–2191, 2020.

[22] X. Huang, J. Zhang, Q. Wu, L. Fan, and C. Yuan, "A coarse-to-fine algorithm for matching and registration in 3d cross-source point clouds," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2965–2977, 2017.

[23] V. Sarode, X. Li, H. Goforth, Y. Aoki, R. A. Srivatsan, S. Lucey, and H. Choset, "Pcrnet: point cloud registration network using pointnet encoding," *arXiv preprint arXiv:1908.07906*, 2019.

[24] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3523–3532.

[25] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "Pointnetlk: Robust & efficient point cloud registration using pointnet," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7163–7172.

[26] H. Liu, H. Yuan, Q. Liu, J. Hou, H. Zeng, and S. Kwong, "A hybrid compression framework for color attributes of static 3d point clouds," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[27] W. Zhu, Z. Ma, Y. Xu, L. Li, and Z. Li, "View-dependent dynamic point cloud compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 2, pp. 765–781, 2020.

[28] Y. Shen, W. Dai, C. Li, J. Zou, and H. Xiong, "Multi-scale structured dictionary learning for 3-d point cloud attribute compression," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[29] O. Dovrat, I. Lang, and S. Avidan, "Learning to sample," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2760–2769.

[30] I. Lang, A. Manor, and S. Avidan, "Samplenet: differentiable point cloud sampling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7578–7588.

[31] W. Zhu, Y. Xu, D. Ding, Z. Ma, and M. Nilsson, "Lossy point cloud geometry compression via region-wise processing," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[32] W. Jia, L. Li, A. Akhtar, Z. Li, and S. Liu, "Convolutional neural network-based occupancy map accuracy improvement for video-based point cloud compression," *IEEE Transactions on Multimedia*, 2021.

[33] H. Liu, H. Yuan, Q. Liu, J. Hou, and J. Liu, "A comprehensive study and comparison of core technologies for mpeg 3-d point cloud compression," *IEEE Transactions on Broadcasting*, vol. 66, no. 3, pp. 701–717, 2019.

[34] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *Proceedings of the International Conference on Learning Representations*, 2017.

[35] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *Proceedings of the International Conference on Learning Representations*, 2018.

[36] J. Wang, D. Ding, Z. Li, and Z. Ma, "Multiscale point cloud geometry compression," in *2021 Data Compression Conference (DCC)*. IEEE, 2021, pp. 73–82.

[37] J. Wang, H. Zhu, H. Liu, and Z. Ma, "Lossy point cloud geometry compression via end-to-end learning," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[38] R. Kimmel and J. A. Sethian, "Computing geodesic paths on manifolds," *Proceedings of the National Academy of Sciences*, vol. 95, no. 15, pp. 8431–8435, 1998.

[39] W. Wu, Z. Qi, and L. Fuxin, "Pointconv: Deep convolutional networks on 3d point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9621–9630.

[40] R. Bridson, "Fast poisson disk sampling in arbitrary dimensions." *SIGGRAPH sketches*, vol. 10, pp. 1 278 780–1 278 807, 2007.

[41] L.-Y. Wei, "Parallel poisson disk sampling," *ACM Transactions on Graphics (TOG)*, vol. 27, no. 3, pp. 1–9, 2008.

[42] F. Shao, Y. Luo, P. Liu, J. Chen, Y. Yang, Y. Lu, and J. Xiao, "Active learning for point cloud semantic segmentation via spatial-structural diversity reasoning," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 2575–2585.

[43] Z. Chen, T. Guan, Y. Luo, Y. Wang, K. Luo, and L. Xu, "Pc-net: A deep network for 3d point clouds analysis," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 465–472.

[44] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European conference on computer vision*. Springer, 2016, pp. 525–542.

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[47] M. Angelina Uy and G. Hee Lee, "Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4470–4479.

[48] Z. Kuang, J. Yu, J. Fan, and M. Tan, "Deep point convolutional approach for 3d model retrieval," in *2018 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2018, pp. 1–6.

[49] D. Zhang, F. He, Z. Tu, L. Zou, and Y. Chen, "Pointwise geometric and semantic learning network on 3d point clouds," *Integrated Computer-Aided Engineering*, vol. 27, no. 1, pp. 57–75, 2020.

[50] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3d point clouds," in *International conference on machine learning*. PMLR, 2018, pp. 40–49.

[51] Y. Sun, Y. Wang, Z. Liu, J. Siegel, and S. Sarma, "Pointgrow: Autoregressively learned point cloud generation with self-attention," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 61–70.

[52] C.-L. Li, M. Zaheer, Y. Zhang, B. Poczos, and R. Salakhutdinov, "Point cloud gan," *arXiv preprint arXiv:1810.05795*, 2018.

[53] G. Elbaz, T. Avraham, and A. Fischer, "3d point cloud registration for localization using a deep neural network auto-encoder," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4631–4640.

[54] Z. J. Yew and G. H. Lee, "3dfeat-net: Weakly supervised local 3d features for point cloud registration," in *European Conference on Computer Vision*. Springer, 2018, pp. 630–646.

[55] A. Zaganidis, L. Sun, T. Duckett, and G. Cielniak, "Integrating deep semantic segmentation into 3-d point cloud registration," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2942–2949, 2018.

[56] Z. Zhang, J. Sun, Y. Dai, D. Zhou, X. Song, and M. He, "End-to-end learning the partial permutation matrix for robust 3d point cloud registration," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 3, 2022, pp. 3399–3407.

[57] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4490–4499.

[58] B. Li, "3d fully convolutional network for vehicle detection in point cloud," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1513–1518.

[59] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 770–779.

[60] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1355–1361.

[61] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.

[62] Y. Chen, Z. Tu, L. Ge, D. Zhang, R. Chen, and J. Yuan, "So-handnet: Self-organizing network for 3d hand pose estimation with semi-supervised learning," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6961–6970.

[63] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and 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.

[64] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3d point set upsampling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5958–5967.

[65] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and 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.

[66] Y. Qian, J. Hou, S. Kwong, and Y. He, "Pugeo-net: A geometry-centric network for 3d point cloud upsampling," in *European Conference on Computer Vision*. Springer, 2020, pp. 752–769.

[67] E. Nezhadarya, E. Taghavi, R. Razani, B. Liu, and J. Luo, "Adaptive hierarchical down-sampling for point cloud classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 956–12 964.

[68] J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian, "Modeling point clouds with self-attention and gumbel subset sampling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3323–3332.

[69] J. Liu, J. Guo, and D. Xu, "Apsnet: Toward adaptive point sampling for efficient 3d action recognition," *IEEE Transactions on Image Processing*, vol. 31, pp. 5287–5302, 2022.

[70] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and 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.

[71] Y. Saad, *Iterative methods for sparse linear systems*. SIAM, 2003.

[72] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and 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.

[73] 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.

[74] Y. Yang, C. Feng, Y. Shen, and 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.

[75] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mâché approach to learning 3d surface generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 216–224.

[76] D. Vlasic, I. Baran, W. Matusik, and J. Popović, "Articulated mesh animation from multi-view silhouettes," in *ACM SIGGRAPH 2008 papers*, 2008, pp. 1–9.

[77] Z. Wu, C. Xiong, C.-Y. Ma, R. Socher, and L. S. Davis, "Adaframe: Adaptive frame selection for fast video recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1278–1287.

[78] S. Mei, M. Ma, S. Wan, J. Hou, Z. Wang, and D. D. F. Feng, "Patch based video summarization with block sparse representation," *IEEE Transactions on Multimedia*, vol. PP, pp. 1–1, 04 2020.

[79] B. Guo, S. R. Gunn, R. I. Damper, and J. D. Nelson, "Band selection for hyperspectral image classification using mutual information," *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 4, pp. 522–526, 2006.

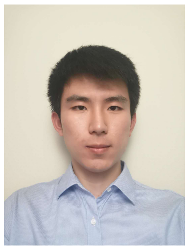
[80] Q. Wang, J. Lin, and Y. Yuan, "Salient band selection for hyperspectral image classification via manifold ranking," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 6, pp. 1279–1289, 2016.

[81] J. Jin, J. Hou, J. Chen, H. Zeng, S. Kwong, and J. Yu, "Deep coarse-to-fine dense light field reconstruction with flexible sampling and geometry-aware fusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

Yue Qian received the B.S. and M.Phil. degrees in Mathematics from The Chinese University of Hong Kong, in 2014 and 2016, and the Ph.D. degree in Computer Science from City University of Hong Kong in 2021. Her research interests include 3D point cloud and geometric processing.

Junhui Hou holds a B.Eng. degree in information engineering (Talented Students Program) from the South China University of Technology (SCUT), Guangzhou, China (2009), an M.Eng. degree in signal and information processing from Northwestern Polytechnical University (NPU), Xi'an, China (2012), and a Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore (2016). He joined the Department of Computer Science at CityU as an Assistant Professor in 2017 and was promoted to Associate Professor in 2023. His research interests are multi-dimensional visual computing.

Dr. Hou was the recipient of several prestigious awards, including the Chinese Government Award for Outstanding Students Study Abroad from China Scholarship Council in 2015 and the Early Career Award (3/381) from the Hong Kong Research Grants Council in 2018. He is an elected member of IEEE MSA-TC, VSPC-TC, and MMSP-TC. He is currently serving as an Associate Editor for *IEEE Transactions on Circuits and Systems for Video Technology*, *IEEE Transactions on Image Processing*, *Signal Processing: Image Communication*, and *The Visual Computer*. He also served as the Guest Editor for the *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* and *Journal of Visual Communication and Image Representation* and an Area Chair of multiple conferences.

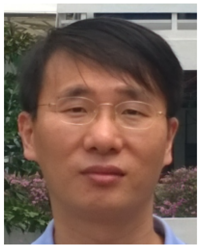


Qijian Zhang received the B.S. degree in Electronic Information Science and Technology from Beijing Normal University, Beijing, China, in 2019. He is currently a PhD student with the Department of Computer Science, City University of Hong Kong. His research interests include computer vision, deep learning, and point cloud processing.

Yiming Zeng received the B.S. degree in automation from South China University of Technology, Guangzhou, China, in 2019. He is currently a PhD student in the department of computer science, city university of Hong Kong, under the supervision of Dr. Junhui Hou. His research interests include deep learning and 3D point cloud processing.

Sam Kwong (Fellow, IEEE) received the B.Sc. degree in electrical engineering from The State University of New York, Buffalo, NY, USA, in 1983, the M.Sc. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1985, and the Ph.D. degree from the University of Hagen, Hagen, Germany, in 1996. From 1985 to 1987, he was a Diagnostic Engineer with the Control Data Canada, Mississauga, ON, Canada. He then joined Bell Northern Research Canada, Ottawa, ON, Canada, as a Member of Scientific Staff, and the Department of Electronic Engineering, City University of Hong Kong (CityU), Hong Kong, as a Lecturer, in 1990. He is currently a Chair Professor with the Department of Computer Science, CityU. His research interests include video coding, pattern recognition, and evolutionary algorithms.

Dr. Kwong is also the Vice-President of Cybernetics, IEEE Systems, Man, and Cybernetics Society. He also serves as an Associate Editor for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and the Journal of Information Science.



Ying He received the BS and MS degrees in electrical engineering from Tsinghua University, China, and the PhD degree in computer science from Stony Brook University. He is currently an Associate Professor with School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests fall into the general areas of visual computing and he is particularly interested in the problems which require geometric analysis and computation.