BIMS-PU: Bi-Directional and Multi-Scale Point Cloud Upsampling

BIMS-PU: Bi-Directional and Multi-Scale Point Cloud Upsampling

Yechao Bai[1], Xiaogang Wang[1], Marcelo H. Ang Jr[1] and Daniela Rus[2]

***Abstract*—The learning and aggregation of multi-scale features are essential in empowering neural networks to capture the fine-grained geometric details in the point cloud upsampling task. Most existing approaches extract multi-scale features from a point cloud of a fixed resolution, hence obtain only a limited level of details. Though an existing approach aggregates a feature hierarchy of different resolutions from a cascade of upsampling sub-network, the training is complex with expensive computation. To address these issues, we construct a new point cloud upsampling pipeline called BIMS-PU that integrates the feature pyramid architecture with a bi-directional up and downsampling path. Specifically, we decompose the up/downsampling procedure into several up/downsampling sub-steps by breaking the target sampling factor into smaller factors. The multi-scale features are naturally produced in a parallel manner and aggregated using a fast feature fusion method. Supervision signal is simultaneously applied to all upsampled point clouds of different scales. Moreover, we formulate a residual block to ease the training of our model. Extensive quantitative and qualitative experiments on different datasets show that our method achieves superior results to state-of-the-art approaches. Last but not least, we demonstrate that point cloud upsampling can improve robot perception by ameliorating the 3D data quality.**

***Index Terms*—Deep Learning for Visual Perception; Computer Vision for Automation.**

I. INTRODUCTION

THE importance of 3D data has become evident in applications like autonomous driving, robotics, medical imaging, etc. Recent studies [1], [2], [3], [4], [5] have shown that point cloud is a compact and efficient 3D representation. However, real-scanned point clouds produced by depth camera and LiDAR are often sparse, noisy, and irregular [6], [7], [8]. As point cloud upsampling can improve the quality of real-scanned data by increasing the point density and uniformity, it has drawn increasing attention in computer vision and robotics community. The upsampled point cloud has to preserve the geometric detail of the underlying surface. To this end, several state-of-the-art point cloud upsampling methods [9], [10], [11], [12] exploit the multi-scale features of the point cloud. PU-Net [9] progressively increases the ball queries of each point to extract multi-scale local region features and concatenates them to generate the upsampled features. Following a similar principle as PU-Net, PU-GCN [11] designs an Inception DenseGCN module, which has parallel DenseGCN branches of different receptive fields, to encode multi-scale context of point clouds. However, the level of geometric detail in the aggregated multi-scale feature is limited as the input resolution is fixed. To get fine-grained details, MPU [10] breaks the upsampling network into successive subnets to progressively upsample the point clouds. Although MPU preserves better details, its back-and-forth conversion between high-dimension feature space and 3D spatial space leads to increased computation complexity and training difficulty. In this work, we adapt the feature pyramid architecture [13], [14], [15], [16] for the point cloud upsampling task and construct a bi-directional and multi-scale upsampling module. Concretely, instead of obtaining the multi-scale feature directly from a feature extractor, our method generates multi-scale point cloud features from a bi-directional up and downsampling pathway inspired by the back projection mechanism [17] developed for image super-resolution [18], [19], [20], [21]. The back projection mechanism uses an iterative up and downsampling procedure to minimize reconstruction errors. Our method decomposes the upsampling/downsampling procedure into sub-steps with a smaller ratio in feature space and generates

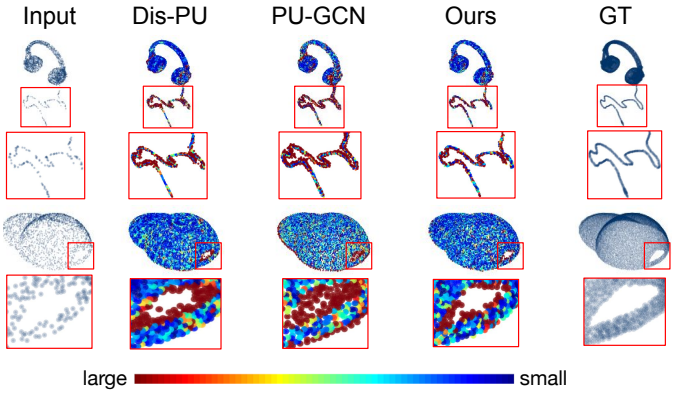


Fig. 1: Qualitative Comparison between our model and state-of-art methods. The color indicates the nearest distance of each output point to the ground truth surface. The result demonstrates that our approach has distinct advantages at challenging places like intersections, the narrow gap between two surfaces, and slender objects. We credited it to the effective multi-scale feature fusion, which enables the model to leverage local and global contextual information.

Manuscript received: February, 22, 2022; Revised May, 18, 2022; Accepted June, 9, 2022.

This paper was recommended for publication by Editor Cesar Cadena upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Research Foundation, Prime Minister's Office, Singapore, under its CREATE program, Singapore-MIT Alliance for Research and Technology (SMART) Future Urban Mobility (FM) IRG. (*Yechao Bai and Xiaogang Wang are co-first authors.*)(*Corresponding author: Yechao Bai.*)

[1]Yechao Bai, Xiaogang Wang and Marcelo H. Ang Jr are with the Department of Mechanical Engineering, National University of Singapore, Singapore. {yechao.bai, xiaogangw}@u.nus.edu, mpeangh@nus.edu.sg

[2]Daniela Rus is with the Massachusetts Institute of Technology, Cambridge, MA, USA. rus@csail.mit.edu

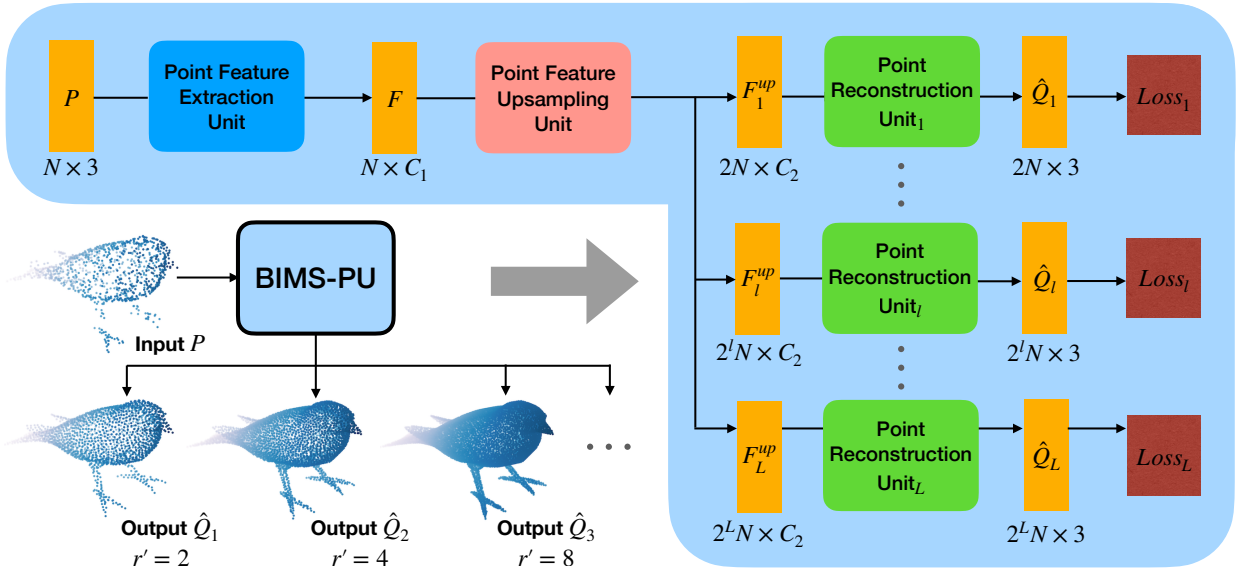


Fig. 2: Overview of the bi-directional multi-scale point cloud upsampling network (BIMS-PU). The architecture of our model has L pathways to upsample a given sparse point cloud P to increasingly denser outputs $Q_1, \dots Q_L$ at the same time, as depicted in the lower left of the figure. Learning a multi-scale point cloud upsampling task enables our model to capture rich geometric details of different levels of resolution.

the multi-scale upsampled point cloud features in a parallel manner. The goal is to decrease the optimization difficulty via decomposing the task into multiple simpler sub-tasks [22]. Next, we leverage a fast weighted feature fusion method to aggregate the resultant multi-scale features. Subsequently, upsampled point cloud features of each scale get reconstructed into 3D point clouds. Supervision signals are applied simultaneously at each output scale. The multi-scale supervision guides the network's training to fuse multi-scale features to be more discriminative. To enable the model to learn complex mapping with additional height and width, we formulate a simple yet highly effective residual block based on the residual learning concept [23] to expand/squeeze the number of channels of a point feature during feature expansion and point reconstruction. The quantitative and qualitative results show that the bi-directional multi-scale up/downsampling pathway improves the fine-grained geometric details of the upsampled point cloud. This is because it enables the up/downsampling operators to be trained with features of different resolutions produced by multi-scale features fusion. Lastly, to verify the value of point cloud upsampling to the robotics community, we design an experiment to show that upsampling is beneficial to point cloud classification, a fundamental robot perception task. In summary, our contributions are:

- Design a bi-directional multi-scale upsampling module;
- Propose training with multi-scale supervision to facilitate the multi-scale feature fusion to produce more discriminative features;
- Conduct extensive quantitative and qualitative experiments on synthetic and real-world datasets to show that our method achieves superior results to state-of-the-art approaches;
- Design an experiment to verify the value of point cloud upsampling to robot perception.

II. RELATED WORKS

Learning-based point cloud upsampling. PointNet [1] and the multi-scale variant PointNet++ [2] propose networks that directly consume point cloud for several 3D recognition tasks. Based on PointNet++, PU-Net [9] designs a point-based network for the point cloud upsampling task. It adopts the hierarchical feature learning mechanism [2] for feature extraction and upsamples point cloud at patch-level. EC-Net [24] designs an edge-aware network and a joint loss to deliberately improve the consolidation near the edge. However, it requires expensive edge annotation for training. To generate outputs of large upsampling factor, MPU [10] progressively upsampled the point cloud to different levels of resolution with a cascade of sub-networks. Unlike previous encoder-decoder networks, PU-GAN [25] incorporates the adversarial training concept into a point upsampling network. It uses a self-attention unit to leverage the long-range context dependencies in the upsampling module. The geometric-centric network, PUGeo-Net [26], explicitly learns the first and second fundamental forms for point cloud upsampling. However, it requires the normals of points as a supervision signal, which is not directly available in a real-scanned point cloud. PU-GCN [11] focuses on improving the upsampling module and the feature extraction module for point cloud upsampling. It integrates the graphical convolutional network into the upsampling module and designs an Inception-based feature extraction module. The recently proposed Dis-PU [12] disentangle the point upsampling tasks into dense point generation and point spatial refinement. To achieve this, they design a network that consists of two cascaded sub-networks. Though both MPU [10] and our methods produce multi-resolution point clouds, our method is not a kind of progressive upsampling method. There are two distinctive differences. First, MPU [10] consists of a cascade of sub-networks, whereas our method produces multi-scale outputs in a single network. Second, the MPU progressively

trains L subnets using $2L+1$ stages, whereas we apply multi-scale supervision signals to each output scale simultaneously, so training can be completed in only one go. PU-GCN [11] and Dis-PU [12] extract local and global features to learn fine-grained details using graphical convolutional network and attention mechanism. However, their method has a high computation demand due to intensive use of kNN and self-attention operation. Our method uses a hierarchical network architecture that is computationally efficient to extract multi-scale features to grasp the fine-grained patterns.

Multi-scale feature representation and aggregation. Multi-scale feature representation and aggregation are one of the main problems in visual perception tasks. Lin et al. [13] proposes a top-down architecture with lateral connections to fuse multi-scale features instead of directly using the pyramidal feature hierarchy for prediction. Taking one step further, Liu et al. [14] adds a bottom-up pathway to enhance the entire feature hierarchy. Ghiasi et al. [15] adopts a neural architecture search to discover a new feature pyramid architecture. Tan et al. [16] revises the architecture design to be more intuitive and principled. The main difference between our work and the pyramid feature architecture in 2D visual perception tasks is that the latter obtains the multi-scale feature directly from feature extractors. But our approach generates a pyramidal point cloud feature from a bi-directional up/downsampling pathway with shortcut connections.

III. APPROACH

A. Overview

Given a sparse point cloud $\mathcal{P}$ of N points, our network outputs L denser point cloud $\{\hat{Q}_1, \ldots, \hat{Q}_l, \ldots, \hat{Q}_L\}$ where $\hat{Q}_l \in \mathbb{R}^{r'N \times 3}$, $r' = 2^l$ is the intermediate upsampling factor and $r = 2^L$ is the desired upsampling factor. Q is the ground truth point cloud with rN points. The upsampled point clouds should lie on the underlying surface of the object and have a uniform distribution. Our network consists of three parts: point feature extraction, point feature expansion and point reconstruction.

Point feature extraction. The feature extractor learns point feature $F \in \mathbb{R}^{N \times C_1}$ from the input point cloud $\mathcal{P} \in \mathbb{R}^{N \times d}$, $C_1 > d$. In this case, the feature of the input is the 3D coordinates of the point cloud, namely $d = 3$. We adopt the feature extractor in MPU [10] which uses dynamic graph convolution [27] to extract point features from local neighborhoods via kNN search in feature space and exploit a dense connection to facilitate information reuse.

Point feature expansion. In this part, we propose a bi-directional multi-scale upsampling module to expand the point features. It takes the point feature F as input and outputs L upsampled point features $\{F_1^{up}, \ldots, F_l^{up}, \ldots, F_L^{up}\}$, where $F_l^{up} \in \mathbb{R}^{r'N \times C_2}$, $F_L^{up} \in \mathbb{R}^{rN \times C_2}$ and $C_2 < C_1$. The bi-directional up and downsampling path preserves the fine-grained geometric details by learning the up/down-sampling operators with global and local context provided by the point features of different resolutions.

Point reconstruction. To reconstruct the L upsampled point features from latent space to coordinate space, we assign each upsampled feature with one 2-layer Multi-Layer Perceptron (MLP) to regress the 3D coordinates.

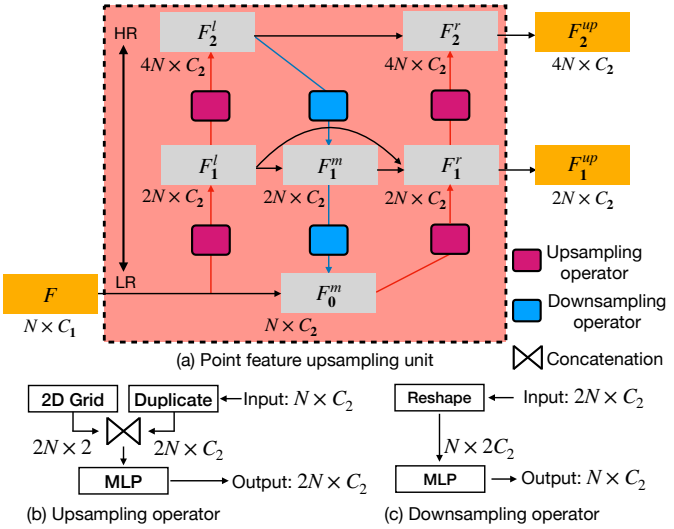


Fig. 3: Illustration of the point feature upsampling unit in Fig. 2. For simplicity, the target upsampling factor is 4.

B. Bi-directional Multi-scale Upsampling Module

The proposed bi-directional multi-scale upsampling module consists of two sets of upsampling operators and one set of downsampling operators as shown in Fig 3(a). The number of operator in each set is L, where $2^L = r$. The scaling factor of each operator is 2. We illustrate the up/downsampling operator in Fig 3(b)/(c). This upsampling operator combines the duplicated point features and a 2D grid [28], [10] that adds spatial variation to help spread out the points. Then it uses shared MLP to produce the upsampled point features. The downsampling operator reshapes the feature and then uses shared MLP to generate the downsampled point features. We leverage hierarchical network architecture which is computationally efficient to learn local/global fine grained patterns instead of using self-attention operator or graphical convolutional network. Given point feature F from feature extraction as input, upsampling module first maps the low-resolution (LR) point feature to a high-resolution (HR) point feature using the first set of upsampling operators on the left side. Then, it maps the HR point feature back to the LR point feature using the set of downsampling operators in the middle. Lastly, the reconstructed LR point feature is mapped to a HR feature point by the second set of upsampling operators on the right side. Our upsampling module uses the shortcut connections and a weighted feature fusion method to achieve a fast and efficient multi-scale feature fusion. For simplicity the desired upsampling factor r is set to 4. Concretely, we describe the two upsampled point feature outputs $F_1^{\text{up}}, F_2^{\text{up}}$, and an intermediate fused feature F_1^m in Fig. 3.

$$
\begin{aligned}
F_2^{\text{up}} &= \frac{w_1 \cdot F_2^l + w_2 \cdot \text{Up}\left(F_1^r\right)}{w_1 + w_2 + \epsilon} \\
F_1^{\text{up}} &= \frac{w_1' \cdot F_1^l + w_2' \cdot F_1^m + w_3' \cdot \text{Up}\left(F_0^m\right)}{w_1' + w_2' + w_3' + \epsilon} \\
F_1^m &= \frac{w_1'' \cdot F_1^l + w_2'' \cdot \text{Down}\left(F_2^l\right)}{w_1'' + w_2'' + \epsilon}
\end{aligned} \tag{1}
$$

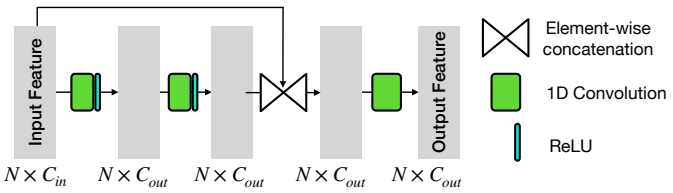


Fig. 4: Illustration of the residual block.

where Up and Down is the up and downsampling operation; w_i is a learnable weight that represents the importance of each input, which is ensured to be positive by applying ReLU [29] after each w_i; ϵ is a small value to avoid numerical instability.

C. Residual Block

In this work, we design a lightweight residual block, as illustrated in Fig. 4, to expand/squeeze the channel of point feature during feature expansion and point reconstruction. Residual learning have proven to be effective at easing the optimization of a network during training [23]. Our intention is to use the residual block to allow the network to learn a complex mapping with increased model complexity namely the scalable bidirectional pathway and cross-scale shortcut links in our model.

D. Multi-scale Supervision.

Our model generates multiple intermediate upsampled point clouds in *one* feed-forward pass. Supervision signals are applied *simultaneously* at each output scale. Notably, we do not downsample the ground truth to create the multi-scale supervision label to avoid potential artificial artifacts and laborious effort. The multi-scale supervision guides the network's training to fuse multi-scale features to be more *discriminative*. Additionally, decomposing the task into multiple simpler sub-tasks decrease the optimization difficulty. It allows our model to have a large representation capacity to learn complicated mappings, thus achieve fewer outliers and render better geometric details. We train our upsampling network with multi-scale supervision using a robust joint loss:

$$\begin{aligned} \mathcal{L} &= \sum_{i=1}^{L} \alpha_i \cdot \mathcal{L}_{\text{joint}}\left(Q, \hat{Q}_i\right) \\ \mathcal{L}_{\text{joint}} &= \mathcal{L}_{\text{CD}} + \lambda \cdot \mathcal{L}_{\text{rep}} \end{aligned} \tag{2}$$

where Q and $\hat{Q}_i$ are the ground truth and multi-scale output point cloud respectively; $\alpha_i, \lambda \in [0, 1]$ are weighting factors; $\mathcal{L}_{\text{CD}}(\cdot)$ means Chamfer Distance [3] which measures the average closest point distance between two point sets; $\mathcal{L}_{\text{rep}}(\cdot)$ means Repulsion Loss [9] which encourage the generated points to distribute more uniformly.

IV. EXPERIMENTS

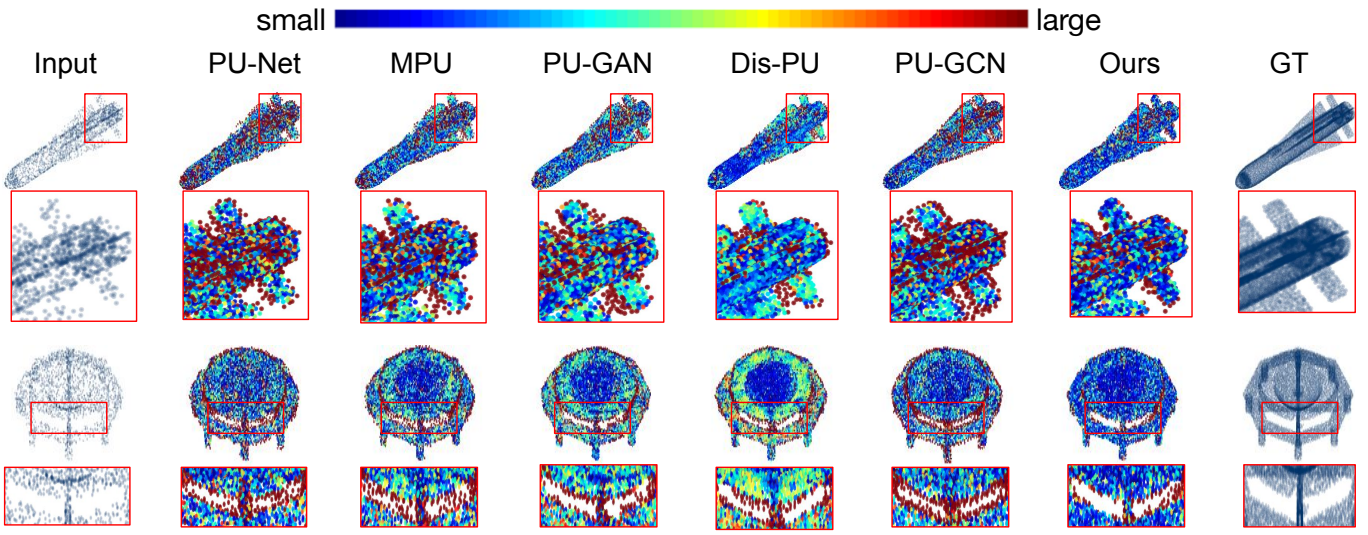
A. Implementation Details

Datasets For quantitative/qualitative comparisons between models, we employ two synthetic datasets and one real-scanned dataset. (1) PU-GAN's dataset provides 120 training and 27 testing objects. (2) PU1K dataset proposed in PU-GCN [11] consists of 1,020 training and 127 testing objects. (3) The real-scanned dataset ScanObjectNN [30] contains 2,902 point cloud objects in 15 categories manually filtered and selected from SceneNN [31] and ScanNet [32]. Each object has 2,048 points. Since the ground truth is not available, we only conduct qualitative experiments on the real-world dataset with pre-trained models. To demonstrate point cloud upsampling is beneficial to 3D object classification, we employ both synthetic and real-scanned datasets in our experiment in Section IV-E. The synthetic dataset is the ModelNet40 [33] dataset, which contains point clouds of 40 common object categories sampled from 100 unique CAD models per category. The real-scanned dataset is ScanObjectNN [30].

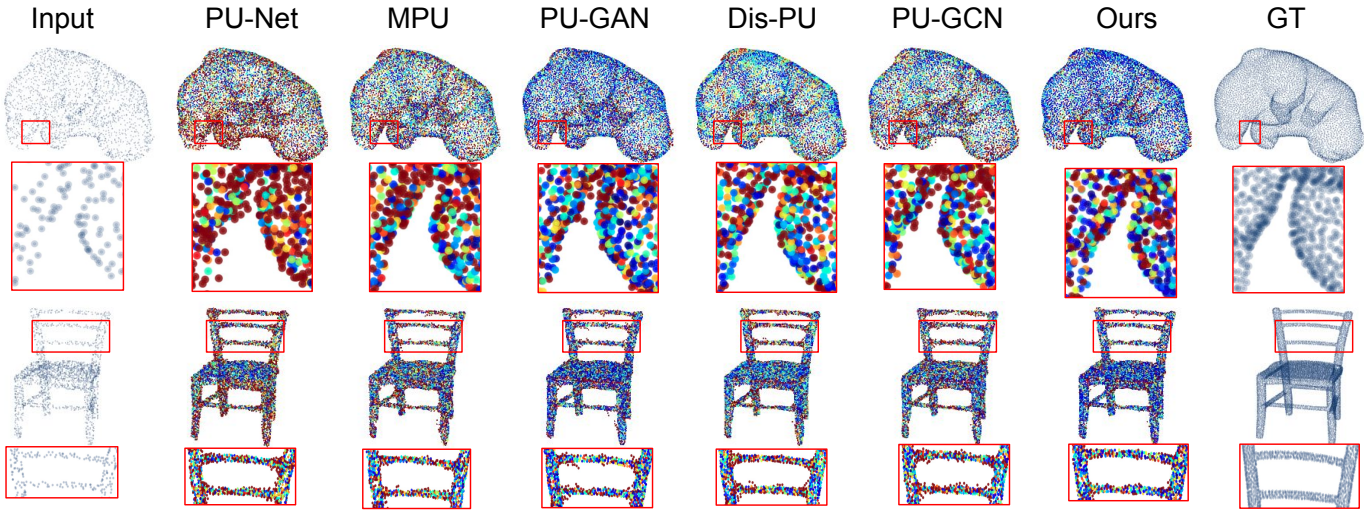
Training details For training, we use the training data provided by PU-GAN's dataset [25] and PU1K [11] and follows the settings in PU-GAN [25] and PU-GCN [11] for model comparison. Concretely, the ground truth point clouds Q has 1,024 points; the input point clouds P of 256 points are randomly downsampled from the ground truth point cloud Q on the fly during training; the upsampling ratio r is 4. We train our model for 400 epochs using the Adam optimizer with an initial learning rate of 0.001. The batch size is 64 on PU1K and 28 on PU-GAN's dataset. We decrease the learning rate by a factor of 0.7 for every 40 epochs. Data augmentation techniques applied includes random rotation/scaling/shifting. In the point feature extraction unit the k for k-nearest neighbors search is 16. C_1 and C_2 in the feature expansion unit are 648 and 128. We use $\alpha_1 = 0.6, \alpha_2 = 1.0$ and $\alpha_1 = 0.6, \alpha_2 = 0.8, \alpha_3 = 1.0$ in Eq. 2 for $r = 4$ and $r = 16$ respectively. We implemented our network using PyTorch and all experiments are conducted on an Nvidia RTX 2080 GPU.

Testing details For testing, we adopt the commonly used patch-based strategy [9], [10], [25], [11], [12] as follows. First, use Poisson disk sampling to generate the ground-truth object point clouds Q of rN points from object mesh and then downsample it to get sparse input point clouds P of N points. Second, apply the farthest point sampling [2] to the input point clouds to get query points and extract overlapping input patches of 256 points around each query point using kNN. Next, feed all input patches to an upsampling model and combine the output overlapping point clouds of 1,024 points to get the dense object point cloud. Lastly, apply farthest point sampling to produce a uniform and dense object point cloud that contains rN points. Both PU-GCN and Dis-PU reported model comparisons results using PU-GAN's dataset. We notice two differences in test settings between them and PU-GAN's in their experiment. (1) PU-GAN [25] and Dis-PU [12] use Monte-Carlo downsampling while PU-GCN uses Poisson downsampling[1] to generate the sparse input point cloud P. (2) PU-GAN [25] and PU-GCN [11] use input point cloud of 2,048 points but Dis-PU [12] uses input point cloud of 1,024 points. Because the Monte-Carlo downsampling generates a realistic and non-uniform point cloud distribution, while the Poisson downsampling produces a uniform point cloud distribution, and the former is also used during training for input point cloud generation. Hence, we follow PU-

[1]https://github.com/guochengqian/PU-GCN/issues/3#issuecomment-888289259



(a) Qualitative results on PU1K dataset



(b) Qualitative results on PU-GAN's dataset

Fig. 5: Qualitative comparisons of point cloud upsampling on synthetic dataset. The color indicates the nearest distance of each output point to the ground truth surface. We can see that our model produces a more accurate reconstruction with fewer red points and preserves better geometric details at challenging areas.

GAN [25] to use the widely used Monte-Carlo sampling. Regarding the number of test input points, we also follow PU-GAN [25]'s setting to use 2,048 points for consistency between model comparisons conducted on PU-GAN's dataset and PU1K dataset.

Evaluation metrics The evaluation metrics are (i) Chamfer distance (CD); (ii) Hausdorff distance (HD) [34]; (iii) point-to-surface distance (P2F). A lower evaluation metric indicates a better performance.

B. Quantitative Comparisons

We conduct comparisons on two datasets PU-GAN's dataset [25] and PU1K [11]. The results are shown in Tables I and II, respectively. The recently proposed PUGeo-Net [26] is not included in the comparison as it requires the accurate normal of point for training, which is not directly available in point clouds.

TABLE I: **Quantitative comparisons with the state-of-the-art on PU-GAN's dataset.** The units of CD, HD, and P2F are 10^{-3}. The best result is highlighted in bold letters and the runner-up is highlighted with an underline.

Methods	4×				16×			
	Size	CD↓	HD↓	P2F↓	Size	CD↓	HD↓	P2F↓
PU-Net [9]	10.1M	0.72	8.94	6.84	24.5M	0.38	6.36	8.44
MPU [10]	23.1M	0.49	6.11	3.96	92.5M	0.19	5.58	3.52
PU-GAN [25]	9.6M	0.28	4.64	2.33	9.6M	0.23	6.09	3.31
PU-GCN [11]	9.7M	<u>0.27</u>	<u>4.38</u>	2.80	9.7M	<u>0.18</u>	<u>4.72</u>	3.15
Dis-PU [12]	13.2M	**0.24**	4.63	<u>2.23</u>	13.2M	**0.16**	8.14	**2.43**
Our	8.3M	0.28	**4.28**	**2.05**	15.6M	**0.16**	**4.70**	<u>2.59</u>

Comparisons on PU-GAN's dataset. Quantitative comparisons between models on PU-GAN's dataset under different upsampling ratios are presented in Table I. The results show that our method performs competitively to state-of-the-art approaches under both small and large upsampling ratios. Notably, our model has the smallest model size when $r = 4$ and is 37% lighter than the runner-up model Dis-PU. Though

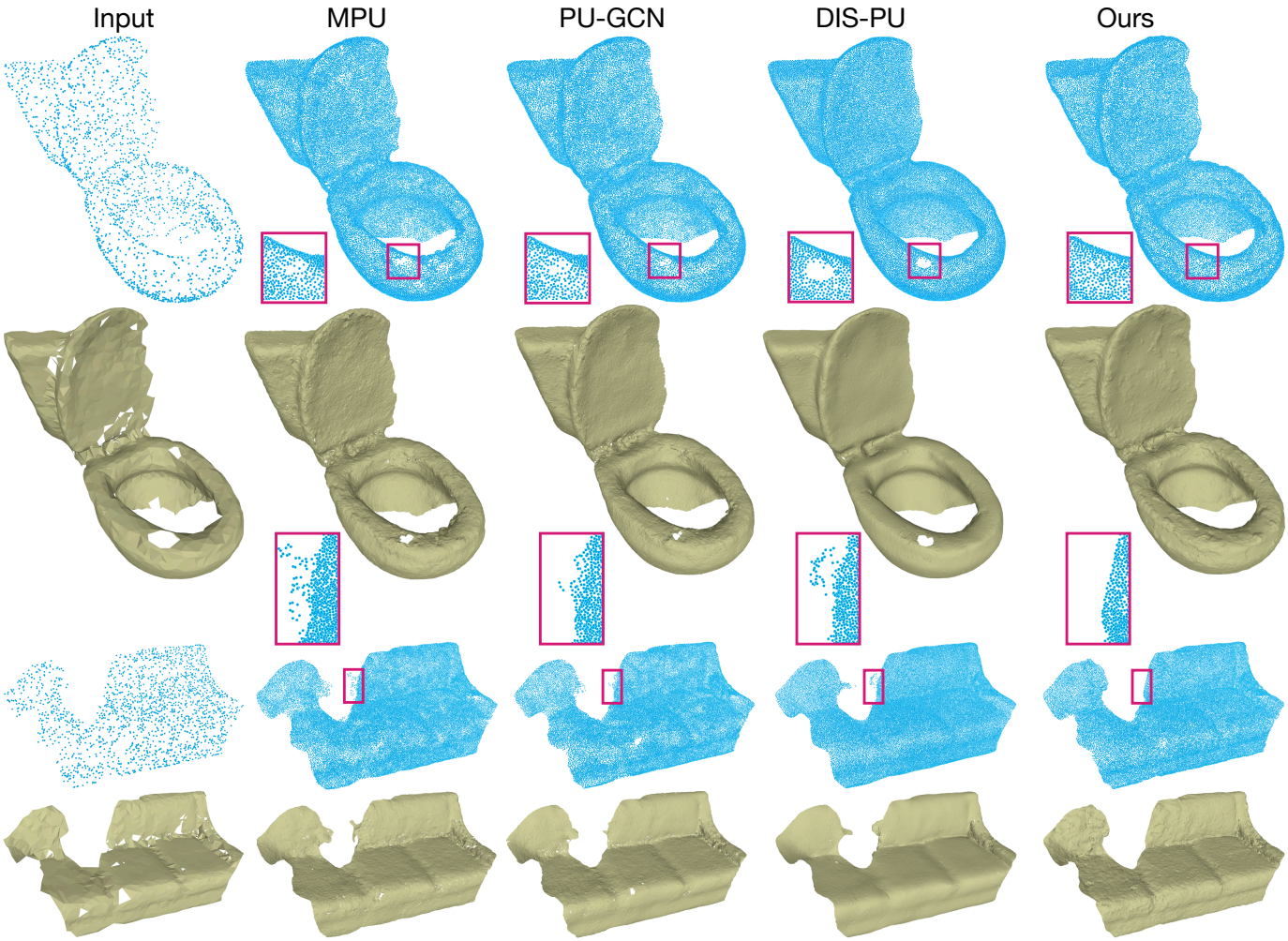


Fig. 6: Qualitative comparison of upsampled (16×) point clouds generated using different methods and its 3D mesh reconstruction. The sparse inputs of 2,048 points are from real-scanned dataset ScanObjectNN [30]. While Dis-PU [12] has an advantage in surface smoothness, our method outperforms others in terms of producing a more accurate reconstruction that has fewer surface defects and outliers.

our model size grows as the upsampling ratio increases, when $r = 16$, it is still comparable to the size of Dis-PU and much smaller than PU-Net and MPU. The input and ground truth test data are generated following PU-GAN's setting when testing $r = 4$ and $r = 16$. We train the models on PU-GAN [25]'s dataset using their released source code and report the test performance using the best model obtained from training. For $r = 4$, we directly use the result of PU-Net, MPU, and PU-GAN from PU-GAN [25]'s paper. Though Dis-PU [12] and PU-GCN [11] have conducted model comparisons using PU-GAN's dataset under $r = 4$ and $r = 16$, we don't use the results in their paper because they used different test settings from PU-GAN [25], which is discussed in Section IV-A-Testing in detail.

Comparisons on PU1K. Quantitative comparisons on PU1K dataset are presented in Table II. We conduct two sets of experiments. One uses test input point cloud generated using Poisson downsampling provided by PU-GCN [11]. The other one uses input point cloud generated using Monte-Carlo downsampling to produce a more realistic and non-uniform distribution. For PU-Net [9], MPU [10], and PU-GCN [11], we use the pre-trained model provided by PU-GCN [11] to get their results. As the pre-trained model of PU-GAN [25] and Dis-PU [12] is not provided, we train the models on PU1K using their released codes and get the result using the best model. In both sets of experiments, our model presents competitive performance. We can see that all models generally performs better when the input point cloud is generated using

TABLE II: **Quantitative comparisons with the state-of-the-art on PU1K.** We conduct two sets of experiments. One uses input point cloud generated using Poisson downsampling as in the paper [11]. Another one uses input point cloud generated using Monte-Carlo downsampling which produces more realistic non-uniform distribution distribution. The units of CD, HD, and P2F are 10^{-3}. The best result is highlighted in bold letters and the runner-up is highlighted with an underline.

Model	Size	Monte-Carlo			Poisson		
		CD↓	HD↓	P2F↓	CD↓	HD↓	P2F↓
PU-Net [9]	10.1M	0.623	10.907	2.877	1.155	15.170	4.834
MPU [10]	23.1M	0.534	9.725	2.286	0.935	13.327	3.551
PU-GAN [25]	<u>9.6M</u>	0.439	<u>7.697</u>	2.117	0.727	9.622	2.936
PU-GCN [11]	9.7M	0.462	7.671	2.125	0.585	<u>7.577</u>	2.499
Dis-PU [12]	13.2M	<u>0.423</u>	**7.095**	<u>1.645</u>	<u>0.550</u>	**6.997**	**2.240**
Our	**8.3M**	**0.420**	7.990	**1.547**	**0.541**	8.360	<u>2.280</u>

TABLE III: **Ablation study.** Experiments are conducted on PU-GAN's dataset. The units of CD, HD and P2F are 10^{-3}. MS is the abbreviation of multi-scale. The effectiveness of our proposed components (residual block, multi-scale supervision, and multi-scale feature fusion) for point cloud upsampling is validated.

Model	Residual	MS Supervision	MS Fusion	CD↓	HD↓	P2F↓
A				0.30	6.05	2.77
B	✓			0.28	5.24	2.17
C	✓	✓		0.28	4.87	2.04
Full	✓	✓	✓	0.28	4.28	2.05

Monte-Carlo downsampling as it is the downsampling method used during training (see Section IV-A-Training details). Our superior performances on the Monte-Carlo setting verified that our model is more robust to non-uniform points. We also get lower errors on CD and P2F, which shows that our model is able to reconstruct more accurate object shapes.

C. Ablation Study

We analyze the contribution of each component of our network on the PU-GAN's dataset in Table III, which includes multi-scale fusion, multi-scale supervision, and residual block. We remove each component from the full model one by one (from bottom to top) and measure the performances in terms of CD, HD and P2F. As shown in Table III, all components contribute to the full model since removing any component hampers the performance.

D. Qualitative Comparisons

We compare our model qualitatively with other methods on two synthetic datasets and one real-word dataset. The results are shown in Fig. 5 and 6. In Fig. 5, the color indicates the nearest distance of each output point to the ground truth surface. We observe three distinct advantages of our approach: 1) Generate points with lower error in the area near the sharp edge, and the edges are cleaner and sharper. 2) Produces fewer outliers in challenging areas like the joint, intersection, and the narrow gap between two surfaces. 3) Preserve better geometric details of slender objects. Specifically, the number of points generally increases along the longitudinal direction of the objects. In Fig. 6, we compare the upsampled point cloud generated using different methods and its 3D mesh reconstruction, the noisy and sparse inputs are from the real-scanned dataset ScanObjectNN [30], where we set $r = 16$. While Dis-PU [12] has an advantage in surface smoothness, our method outperforms others in terms of producing a more accurate reconstruction that has fewer surface defects and outliers. The qualitative results suggest that our model possesses a better understanding of the global and local context relationship and is capable of generating high-fidelity object details.

E. Benefit of upsampling to point cloud classification

Point cloud object classification is a fundamental task to robot perception which is crucial to downstream tasks like object detection and semantic segmentation. To demonstrate the value of point cloud upsampling to the robotic community, we design an experiment to show that upsampling is beneficial to point cloud object classification. In this experiment, We employ a synthetic dataset ModelNet40 [33] and a real-scanned dataset ScanObejctNN [30] and choose two widely used models PointNet++[2] [2] and PointNet[3] [1] to perform point cloud shape classification on ModelNet40 and ScanObjectNN respectively. The PointNet++ is pre-trained on ModelNet40, whereas PointNet is pre-trained on ScanObjectNN's hardest variant PB_T50_RS. First, we use farthest point sampling and random sampling to sample the testing point clouds to point clouds of 512 points as ground truth data and point clouds of 128 points as input data. Next, we upsample the input point clouds by four times using a set of point cloud upsampling models. The upsampling models are pre-trained on PU-GAN's [25] dataset used in Table I. We compare the classification accuracy of the ground-truth, randomly-downsampled, and upsampled point clouds in Table IV. The results show that applying point cloud upsampling to sparse and nonuniform point clouds to generate denser point clouds effectively improves the classification result in both synthetic and real-scanned data. Interestingly the classification performance improvement is more significant on real-scanned data. This demonstrates that the upsampling models can generate new points according to the distribution pattern of the underlying surface. Further, the quantitative classification accuracy comparison indicates the superiority of our model in reconstruction accuracy. Experiments could be designed following a similar principle to show the effect of point cloud upsampling on semantic segmentation and part segmentation.

TABLE IV: **Classification accuracy comparison.** We use random sampling and farthest point sampling on the testing point cloud in synthetic dataset ModelNet40 [33] and real-scanned dataset ScanObjectNN [30] to generate point clouds of 128 points as input and point clouds of 512 points as ground truth. Then we upsample the input point cloud by 4 times using a set of upsampling models. The classification accuracy comparison is conducted between input/upsampled/ground-truth point cloud. The results indicate that point cloud upsampling is beneficial to point cloud classification, and our model is superior in generating new points that reflect the underlying surface of point clouds. Overall and average class accuracy are shown in %.

Model	ModelNet40 [33]		ScanObjectNN [30]	
	OA.	Cls Acc.	OA	Cls Acc.
PU-GAN [25]	84.2	80.0	71.5	67.4
PU-GCN [11]	84.6	80.6	70.9	67.0
Dis-PU [12]	85.0	80.4	71.1	66.8
Ours	**87.1**	**82.5**	**72.0**	**68.2**
Input	79.4	74.1	61.8	56.3
Ground truth	92.5	88.7	74.9	70.6

V. CONCLUSION

In this work, we propose a bi-directional multi-scale upsampling approach for 3D point cloud upsampling. We decompose a bi-directional up/downsampling pathway into sub-up/downsampling steps of smaller scaling factors to produce

[2]https://github.com/yanx27/Pointnet_Pointnet2_pytorch
[3]https://github.com/hkust-vgd/scanobjectnn

a pyramidal multi-scale point feature hierarchy. The point features in the hierarchy are fused and reconstructed to point clouds of different resolutions. Supervision signals are applied to each output point cloud to ensure that the feature fusion produces discriminative features. A simple yet effective residual block is proposed to reduce the optimization difficulty. Extensive quantitative and qualitative results on synthetic and real-world datasets demonstrate that our method achieves superior results compared to state-of-the-art approaches. We demonstrate that point cloud upsampling can improve robot perception by ameliorating the 3D data quality using a simple experiment.

REFERENCES

[1] Qi, Charles R and Su, Hao and Mo, Kaichun and Guibas, Leonidas J, "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. 1, 2, 7

[2] Qi, Charles R and Yi, Li and Su, Hao and Guibas, Leonidas J, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *arXiv preprint arXiv:1706.02413*, 2017. 1, 2, 4, 7

[3] Fan, Haoqiang and Su, Hao and Guibas, Leonidas J, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 605–613. 1, 4

[4] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," *arXiv preprint arXiv:2012.09164*, 2020. 1

[5] X. Wang, M. H. Ang Jr, and G. H. Lee, "Cascaded refinement network for point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 790–799. 1

[6] M. Atzmon, H. Maron, and Y. Lipman, "Point convolutional neural networks by extension operators," *arXiv preprint arXiv:1803.10091*, 2018. 1

[7] P. Hermosilla, T. Ritschel, P.-P. Vázquez, À. Vinacua, and T. Ropinski, "Monte carlo convolution for learning on non-uniformly sampled point clouds," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–12, 2018. 1

[8] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 87–102. 1

[9] Yu, Lequan and Li, Xianzhi and Fu, Chi-Wing and Cohen-Or, Daniel and Heng, Pheng-Ann, "Pu-net: Point cloud upsampling network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2790–2799. 1, 2, 4, 5, 6

[10] 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/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5958–5967. 1, 2, 3, 4, 5, 6

[11] G. Qian, A. Abualshour, G. Li, A. Thabet, and B. Ghanem, "PU-GCN: Point cloud upsampling using graph convolutional networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 683–11 692. 1, 2, 3, 4, 5, 6, 7

[12] R. Li, X. Li, P.-A. Heng, and C.-W. Fu, "Point Cloud Upsampling via Disentangled Refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 344–353. 1, 2, 3, 4, 5, 6, 7

[13] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125. 1, 3

[14] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8759–8768. 1, 3

[15] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Nas-fpn: Learning scalable feature pyramid architecture for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7036–7045. 1, 3

[16] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790. 1, 3

[17] M. Irani and S. Peleg, "Motion analysis for image enhancement: Resolution, occlusion, and transparency," *Journal of visual communication and image representation*, vol. 4, no. 4, pp. 324–335, 1993. 1

[18] S. Dai, M. Han, Y. Wu, and Y. Gong, "Bilateral back-projection for single image super resolution," in *2007 IEEE International Conference on Multimedia and Expo*. IEEE, 2007, pp. 1039–1042. 1

[19] M. Haris, G. Shakhnarovich, and N. Ukita, "Recurrent back-projection network for video super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3897–3906. 1

[20] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1664–1673. 1

[21] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, "Feedback network for image super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3867–3876. 1

[22] F. Zhang, Y. Chen, Z. Li, Z. Hong, J. Liu, F. Ma, J. Han, and E. Ding, "ACFNet: Attentional class feature network for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6798–6807. 2

[23] 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. 2, 4

[24] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "EC-Net: an edge-aware point set consolidation network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 386–402. 2

[25] 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/CVF International Conference on Computer Vision*, 2019, pp. 7203–7212. 2, 4, 5, 6, 7

[26] 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. 2, 5

[27] 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. 3

[28] 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. 3

[29] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Icml*, 2010. 4

[30] Uy, Mikaela Angelina and Pham, Quang-Hieu and Hua, Binh-Son and Nguyen, Thanh and Yeung, Sai-Kit, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1588–1597. 4, 6, 7

[31] Hua, Binh-Son and Pham, Quang-Hieu and Nguyen, Duc Thanh and Tran, Minh-Khoi and Yu, Lap-Fai and Yeung, Sai-Kit, "Scenenn: A scene meshes dataset with annotations," in *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016, pp. 92–101. 4

[32] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839. 4

[33] 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. 4, 7

[34] M. Berger, J. A. Levine, L. G. Nonato, G. Taubin, and C. T. Silva, "A benchmark for surface reconstruction," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 2, pp. 1–17, 2013. 5