RFNet: Recurrent Forward Network for Dense Point Cloud Completion

Tianxin Huang[1] Hao Zou[1] Jinhao Cui[1] Xuemeng Yang[1] Mengmeng Wang[1] Xiangrui Zhao[1]
Jiangning Zhang[1] Yi Yuan[3] Yifan Xu[3] Yong Liu[1,2*]
[1]Zhejiang University [2]Huzhou Institute of Zhejiang University [3]NetEase Fuxi AI Lab

Abstract

Point cloud completion is an interesting and challenging task in 3D vision, aiming to recover complete shapes from sparse and incomplete point clouds. Existing learning-based methods often require vast computation cost to achieve excellent performance, which limits their practical applications. In this paper, we propose a novel Recurrent Forward Network (RFNet), which is composed of three modules: Recurrent Feature Extraction (RFE), Forward Dense Completion (FDC) and Raw Shape Protection (RSP). The RFE extracts multiple global features from the incomplete point clouds for different recurrent levels, and the FDC generates point clouds in a coarse-to-fine pipeline. The RSP introduces details from the original incomplete models to refine the completion results. Besides, we propose a Sampling Chamfer Distance to better capture the shapes of models and a new Balanced Expansion Constraint to restrict the expansion distances from coarse to fine. According to the experiments on ShapeNet and KITTI, our network can achieve the state-of-the-art with lower memory cost and faster convergence.

1. Introduction

With the rapid development of real-time 3D sensors like LiDAR and depth camera, 3D data has attracted more and more attention in computer vision and robotics. As a representation which describes the scene better than 2D images, 3D point clouds have been widely used in applications such as SLAM [1] and object detection [6, 18, 22]. However, point clouds acquired from sensors are often incomplete and sparse due to the limitation of resolution and occlusion. As a consequence, recovering complete and high-resolution models from incomplete inputs has been an important and challenging task, known as the point cloud completion.

Since the work of PCN [32], many deep learning based researches have been explored on the 3D point cloud completion work. Some of them are based on 3D grids and 3D convolution neural networks(CNNs), such as GRNet [28]. The others are built on the structure of PointNet [19] and PointNet++ [20], such as TopNet [24] and SANet [27]. These networks need high-dimensional global features or multiple local features to acquire enough shape information from the inputs. Most of them have plenty of parameters and suffer from great memory cost to reach good performance. To address these problems, we propose a novel well-performed recurrent forward point cloud completion framework with designed lightweight modules and parameter-shared operations to reduce the parameters and memory cost. Besides, most works above pay less attention to details of incomplete point clouds, which will cause a distortion of the outputs. Large distortions will lead to meaningless completion results. On this occasion, we merge original shapes from incomplete models with outputs of different resolutions to prevent our completion results from large distortions.

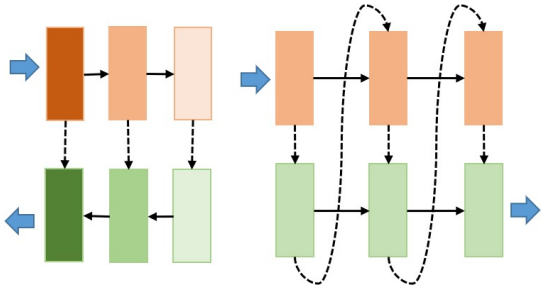

Figure 1. Comparison of U-Net (left) and our framework (right). Rectangles denote operations in networks, while same colors mean same parameters. Our framework recurrently send the coarse completed results and incomplete input to next level, while parameters are shared in some layers.

In this paper, we propose a novel approach named RFNet. As shown in Fig. 1, it is organized in a "forward" framework different from the "backward" framework like U-Net [21], which has been proved working well on segmentation [17, 3] and point cloud completion [27]. However, U-Net is computationally expensive to search and aggregate local features of different resolutions, especially for dense point clouds. In our framework, operations are organized in multiple recurrent levels, which can be divided

* means the corresponding author

into three modules: Recurrent Feature Extraction (RFE), Forward Dense Completion (FDC) and Raw Shape Protection (RSP). In the RFE, we do not extract high-dimensional global features or local features for completion, but leverage multiple short global features in different recurrent levels to decrease the computational cost. We introduce the FDC to generate completed point clouds of different resolutions. The FDC generates the results by creating an initial model and lifting it to high resolutions with a larger lifting ratio than most previous methods. Therefore, our network needs fewer lifting levels to generate dense point clouds. Some layers in the RFE and FDC are parameter-shared to further reduce the model size. We propose the RSP to preserve details of the original incomplete model. Points from the FDC are driven towards their nearest neighbors in original point clouds in the RSP. The driving distances can be controlled by a learnable parameter. In this way, we can preserve more information of the original models. To better capture the shapes and improve the uniformity of results, we apply Chamfer Distance in randomly divided subsets of dense point clouds, named Sampling Chamfer Distance. Besides, we improve the generation continuity in the FDC by constraining expansion distances balanced with their estimated expectation.

The contributions can be summarized as follows:

1. We propose a novel point cloud completion network based on a new recurrent forward framework. In addition to the improvement of the completion result, the memory cost is greatly reduced benefit from this structure;

2. We propose a Raw Shape Protection module to preserve original shapes in a learnable way;

3. We propose a Sampling Chamfer Distance to better capture the shape differences between models and a new Balanced Expansion Constraint to restrict the expansion distances from coarse to fine;

4. The experiments on ShapeNet and KITTI demonstrate that our network outperforms existing methods on the 3D completion task.

2. Related Work

Point Cloud Learning. Early works [5, 9, 14] usually apply 3D CNNs based on voxel representation of 3D point clouds. However, 3D volumes can not be directly acquired. Converting point clouds to 3D voxels is expensive, which also leads to quantization errors caused by ignoring some details of the original data. So Qi *et al*. first introduced a point-based point cloud learning network named PointNet [19]. It processes point clouds with multilayer perceptrons (MLPs) and aggregates features with symmetric functions (*e.g*. max pooling). PointNet++ [20] captures local features by recurrently applying PointNet in local regions acquired by ball query around sampled points. Lots of works have been proposed based on PointNet and PointNet++ such as the point cloud analysis [11, 13, 4, 23], reconstruction [10, 25, 29] or upsampling [31, 15].

Point Cloud Completion. PCN [32] is the first point-based model for point cloud completion. It generates complete model in a two-stage process which first generates a coarse result by a fully-connected network and refines it to a higher resolution with a folding-based network. PFNet [12] completes models by generating missing parts with proposed Point Pyramid Decoder (PPD), which is interesting but may have difficulty generating missing regions with unknown points numbers. TopNet [24] explores the hierarchical rooted tree structure as decoder to generate arbitrary grouping of points in completion task. SANet [27] adopts a commonly used U-Net structure with the skip-connection and self-attention modules to complete missing features. It performs well on the datasets with sparse points. Cascaded refinement network (CANet) [26] adds a cascaded refinement module to achieve transformation from coarse to fine by multiple lifting with a small upsampling ratio. CANet performs better than PCN and TopNet on ShapeNet [2], while it requires for shape priors to complete models. Morphing and sampling network (MSN) [16] improves completion performance by assembling the incomplete models with outputs by minimum density sampling. It also proposes an approximation for the Earth Mover Distance to train the network. GRNet [28] and NSFA [33] achieve completion with quite different ideas. GRNet transforms the incomplete models to 3D grids representation and adopts 3D CNN to learn features and complete models. NSFA treats point cloud completion as upsampling. The hierarchical feature learning architecture in PU-Net [31] is adopted to extract local features. Local features of different resolutions are used to construct points of preserving or missing parts.

3. Methodology

Our RFNet aims to complete the incomplete point clouds to dense shapes with fewer parameters and less cost. As indicated in Fig. 2, the structure is organized in multiple recurrent levels. The output model from the former level is concatenated with the incomplete model and fed to the latter level as a "new incomplete model". The RFE extracts features for completion in different recurrent levels. The FDC creates models of different resolutions based on the output features. Subsequently, shape details from the incomplete model are added to the model by the RSP. The output of the last recurrent level is taken as the final output.

3.1. Recurrent Feature Extraction

Recurrent Feature Extraction module (RFE) involves the Encode Cells and Recover Cells. An Encode Cell extracts output features for completion and state features for the Encode Cell in the next recurrent level. Recover Cells recover the output features by further aggregating the information

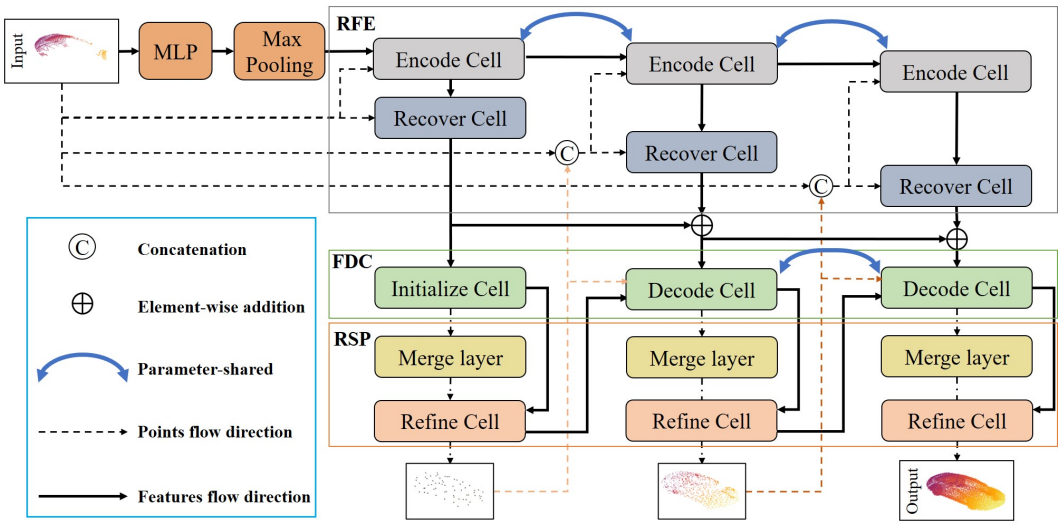

Figure 2. Structure of the RFNet. It is organized in three recurrent levels, which can also be divided into three modules: RFE, FDC, RSP. The output from the former level is concatenated with the incomplete model and fed to the latter level as a "new incomplete model".

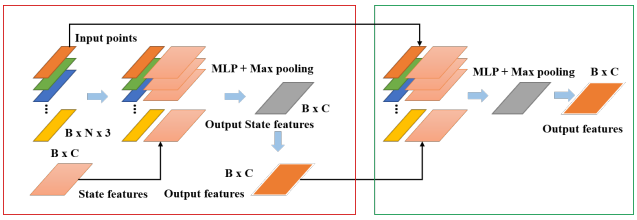

Figure 3. Structures of the Encode Cell and Recover Cell.

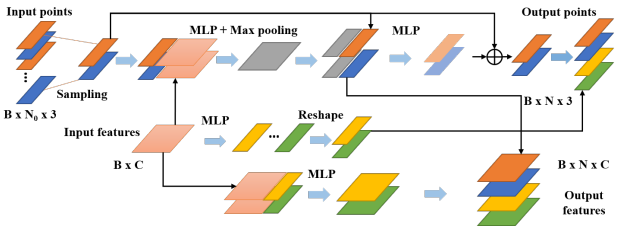

Figure 4. Structure of the Initialize Cell. Generated and sampled points make up the output.

from incomplete models and return the final features for the FDC. The structures of the Encode Cell and Recover Cell are presented in Fig. 3.

Encode Cell. We design Encode Cell to extract a initial feature from the input points, which is parameter-shared between different levels to reduce the parameters. State features are extracted and adopted to help Encode Cell to focus on different regions of input point clouds adaptively.

Recover Cell. The features directly extracted by the Encode Cells are not enough for completion due to the lack of information in the original model. In this condition, Recover Cells are adopted to further aggregate the information from the incomplete model to complete the features extracted by the Encode Cells. In a Recover Cell, input features from the Encode Cell are concatenated with the input points again and fed to MLPs to get the recovered features.

3.2. Forward Dense Completion

Forward Dense Completion module (FDC) including an Initialize Cell and the Decode Cells, generates completed point clouds based on the features from the RFE. The Initialize Cell generates an initial model in the first level, while Decode Cells lift the model to higher resolutions.

Initialize Cell. We design the Initialize Cell to create a basic structure of the completion result. We name the output of the Initialize Cell as "seeds" of the completion result. Because more detailed models with high resolutions are grown from them like trees. Generating seeds directly by a fully-connected network or FoldingNet [30] is a commonly used operation. It is flexible but greatly limited by the effectiveness of a network, having difficulty in constructing complex and non-manifold shapes which cannot be constructed by bending 2D planes. To acquire more accurate seeds, we introduce the contour of the original incomplete model by adding sampled points from the incomplete model. As presented in Fig. 4, half of seeds are acquired through sampled points with predicted offsets, and the others are through network output directly. Though output of the network is limited, it is important because the sampled points are restricted by the incomplete model. The fully-connected network output has satisfactory performance generating missing parts not covered by the incomplete model.

Decode Cell. A Decode Cell is designed to learn a parameter-shared transformation from lower resolutions to K times larger as shown in Fig. 5, which is achieved by predicting K offsets for each point from last level. So, we need state features to record the state and level of current Decode

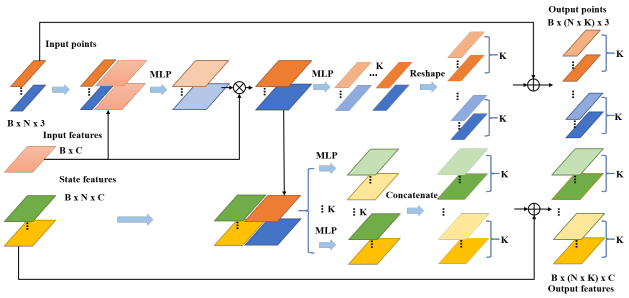

Figure 5. Structure of the Decode Cell. $\bigotimes$ denotes element-wise multiplication. Inputs points and state features are lifted to K times larger with MLPs.

Cell. Input features extracted from Encode and Recover Cell are adopted to introduce the information from original point clouds. K parameter-separate MLPs are used for the prediction of output state features, while a fully connected network is adopted to predict offsets for output points.

3.3. Raw Shape Protection

Raw Shape Protection module (RSP) is consist of Merge Layers and Refine Cells. A Merge Layer introduces details from the original incomplete model to the completed output from the FDC, while a Refine Cell further adjusts the merged result to improve the completion performance.

Merge Layer. A Merge Layer is used to prevent the completed results from going too far from the shapes of the original incomplete models. It will introduce information from the original models to outputs of different levels. In a Merge Layer, points near the incomplete model will be driven towards their nearest neighbors in the incomplete point clouds. In this way, a network output will be merged with details from the incomplete model. The operation of a Merge Layer can be described as follows:

$$dis = \min_{x \in P_0, \forall y \in P} \|x - y\|_2, \tag{1}$$

$$idx = \arg\min_{x \in P_0, \forall y \in P} \|x - y\|_2, \tag{2}$$

$$\hat{P} = e^{-\frac{dis}{\sigma}} P_0(idx) + (1 - e^{-\frac{dis}{\sigma}})P, \tag{3}$$

where σ is a trainable parameter. It learns to acquire the best radius to merge points with their nearest neighbors in the original model. P_0 is the incomplete point set and P is the result of the FDC. $\hat{P}$ is the output of a Merge Layer. The visualization of the Merge Layer is indicated in Fig. 6.

Refine Cell. We add a Refine Cell behind the Merge Layer to fine-tune the positions of points, as shown in Fig. 7.

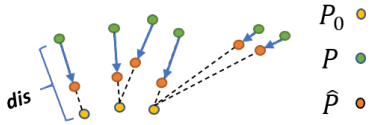

Figure 6. Visualization of the Merge Layer.

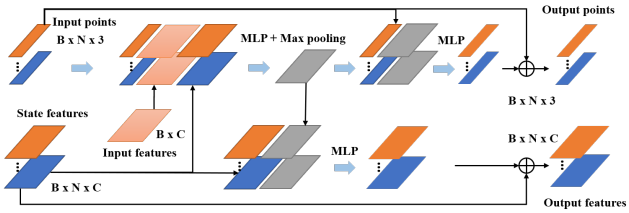

Figure 7. Structure of the Refine Cell. Input points and state features are adjusted based on input features.

Input points, state features and the input feature are concatenated together and combined into a global feature including all information in current level, which will guide the tuning of points and state features.

3.4. Loss function

Basic multilevel loss. To accelerate convergence, we merge the losses of different outputs together to acquire the basic multilevel loss. There are two commonly used loss functions to measure the differences between two point clouds: Chamfer Distance (CD) and Earth Mover Distance (EMD) [7]. Their basic forms are shown as follows:

$$\mathcal{L}_{CD}(S_1, S_2) = \frac{1}{2}\Big(\frac{1}{|S_1|}\sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2 + \frac{1}{|S_2|}\sum_{x \in S_2} \min_{y \in S_1} \|x - y\|_2\Big), \tag{4}$$

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

where S_1 and S_2 are two point sets. CD works mainly on the contours of models, which may lead to a non-uniform result. We propose the Sampling Chamfer Distance (SCD) to improve uniformity. It can be described as follows:

$$\mathcal{L}_{SCD}(S_1, S_2) = \frac{1}{N}\sum_{S_1^i \in D_1, S_2^i \in D_2} \mathcal{L}_{CD}(S_1^i, S_2^i), \tag{6}$$

while $D_1 = RD(S_1, N), D_2 = RD(S_2, N)$. $RD(S, N)$ means to randomly divide the point set S into N isometric sets. l is defined as the number of recurrent levels with low resolutions, while h is that with high resolutions of more than 10000 points. In our work, we apply EMD to the outputs of low recurrent levels as the basic multilevel loss. Due to the high calculation cost of EMD on dense point clouds, we use SCD and CD for our high-resolution outputs. Finally, the basic multilevel loss can be described as follows:

$$\mathcal{L}_{BM} = \sum_{i=1}^{l} \mathcal{L}_{EMD_i} + \sum_{j=l}^{h} (\mathcal{L}_{SCD_j} + \mathcal{L}_{CD_j}). \tag{7}$$

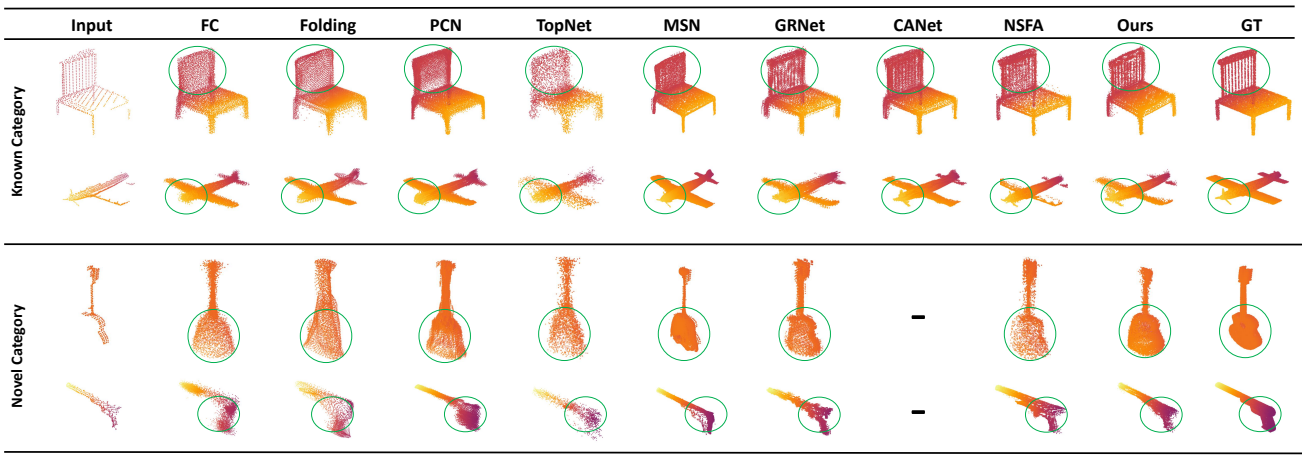

Figure 8. Qualitative comparisons with state-of-the-art methods on ShapeNet. There is no result for CANet on novel category models because there is no known shape prior feature for them.

Balanced Expansion Constraint. Balanced Expansion Constraint is used to prevent points predicted by the Decode Cells from going too far from centers. It will ensure a Decode Cell to generate continuous local shapes instead of discrete points in the 3D space. However, constraining distance from centers directly is too ambiguous because the gradient is zero only when generated points coincide with the corresponding centers. It will inevitably impact the network. On this occasion, we propose the Balanced Expansion Constraint, which can be described as follows:

$$\mathcal{L}_{EC} = \frac{1}{|\hat{S}|} \sum_{x \in \hat{S}, f: S \to \hat{S}} \|x - f^{-1}(x)\|_2, \tag{8}$$

$$E(\mathcal{L}_{EC}) \approx \frac{1}{|\hat{S}_0|} \sum_{x \in \hat{S}_0} \min_{y \in S_0} \|x - y\|_2, \tag{9}$$

$$\mathcal{L}_{BEC} = ReLU(\mathcal{L}_{EC} - \epsilon * E(\mathcal{L}_{EC})), \tag{10}$$

where S and $\hat{S}$ are inputs and outputs of the Decode Cell, respectively, S_0 and $\hat{S}_0$ are the ground truths for them. We consider that the expansion distances are related to the differences between two different resolution models. We estimate expectation of the expansion distances by Equation 9, while using it as an additional item to balance the expansion constraint. In this way, the gradient can be zero when the expansion distances are small enough, eliminating the disturbance to the network. The influence of the additional item can be adjusted by ϵ.

Merge range constraint. Merge range constraint is used to restrain the search radius of a Merge Layer. ξ is a parameter guided by the annealing strategy. It will decrease gradually as iterations increase. In this way, the merge range is small to pay more attention to generation result at the beginning of the iterations, big to introduce more information from the original incomplete model at the later period of iterations. The merge range constraint can be formulated as

$$\mathcal{L}_{MR} = \xi * \|\delta\|_2^2. \tag{11}$$

Overall loss. With a Balanced Expansion Constraint and a merge range constraint for each recurrent level, overall loss is the weighted sum of mentioned losses as follows:

$$\mathcal{L} = w_1 \mathcal{L}_{BM} + w_2 \sum_{i=1}^{n-1} \mathcal{L}_{BEC_i} + w_3 \sum_{j=1}^{n} \mathcal{L}_{MR_j}, \tag{12}$$

where n is defined as the number of recurrent levels, w_1, w_2 and w_3 are weights for different constraints.

4. Experiments

4.1. Dataset and Implementation Details

ShapeNet. ShapeNet [2] for completion contains 30974 models from 8 categories, which is given by PCN [32]. The ground truth models contain 16384 points uniform sampled on the surfaces of mesh models. The partial point clouds are generated by back-projecting 2.5D depth images into 3D. For a fair comparison, we use the same splits as PCN [32].

KITTI. To further test our network, we evaluate it on the real-world scans from KITTI [8]. Cars are acquired with the ground truth object bounding boxes from each frame. The test set includes 2401 partial point clouds labelled as cars.

Metrics. In our work, we adopt the Chamfer Distance (CD) mentioned in Section 3.4 as a global metric for completion performance. However, models may be changed a lot and lose details of the incomplete models during completion, while keeping a small global error. An example is shown in Fig. 9. CD metric of the severely distorted FoldingNet result is even smaller than our network, while its Fidelity error (FD) [32] is more than 6 times larger than ours. In terms of this problem, we use FD as an supplementary evaluation for the distortions. It is defined as the average

Method	FC [32]		Folding [32]		PCN [32]		TopNet [24]		MSN [16]		CANet [26]		GRNet [28]		NSFA [33]		Ours	
Metric	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD
airplane	5.69	5.49	5.96	6.60	5.50	5.14	5.85	7.97	5.60	3.22	**4.79**	2.80	6.44	3.70	5.22	3.37	4.91	**1.98**
cabinet	11.02	9.27	10.83	8.89	10.63	7.28	10.78	12.44	11.90	6.42	**9.97**	4.89	10.39	6.55	10.51	4.94	9.98	**3.49**
car	8.77	10.10	9.27	11.42	8.69	9.47	8.84	10.76	10.70	**6.19**	**8.31**	7.20	9.45	7.77	9.00	7.42	8.66	6.96
chair	10.98	9.86	11.24	10.43	10.99	7.99	10.80	13.50	10.60	4.96	9.49	4.06	9.41	5.30	9.33	4.11	**9.14**	**2.83**
lamp	11.13	10.39	12.17	11.98	11.33	8.75	11.15	13.94	10.70	3.65	8.94	4.15	7.96	4.50	8.26	3.57	**7.16**	**3.02**
sofa	11.75	8.89	11.63	9.25	11.67	7.27	11.41	12.32	11.80	6.04	10.69	3.83	10.50	4.90	10.74	4.17	**10.45**	**2.95**
table	9.32	9.64	9.45	10.16	8.59	8.05	8.79	12.15	8.71	5.38	7.81	4.05	8.44	5.88	7.78	3.99	**7.45**	**2.86**
vessel	9.72	8.53	10.02	10.02	9.66	7.44	9.17	10.63	9.48	4.57	8.05	3.63	8.04	3.93	7.66	3.44	**7.28**	**2.75**
Average	9.79	9.02	10.07	9.85	9.63	7.67	9.60	11.71	9.96	5.05	8.51	4.33	8.83	5.32	8.55	4.38	**8.13**	**3.35**

Table 1. Quantitative Comparisons on known categories of ShapeNet with state-of-the-art methods with the metrics multiplied by 10^3. The bold and underlined values are the best and the second best values, respectively.

Method	FC	Folding	PCN	TopNet	MSN	CANet	GRNet	NSFA	Ours
airplane	5.75	6.08	5.46	7.33	5.59	**4.71**	6.42	5.21	5.01
cabinet	11.10	10.94	10.49	13.20	11.94	**9.93**	10.34	10.49	10.00
car	8.70	8.22	8.50	10.45	10.76	**8.31**	9.56	8.94	8.72
chair	10.98	11.24	10.99	10.80	10.60	9.38	9.38	9.32	**9.31**
lamp	11.77	12.66	11.49	14.17	10.73	8.86	7.77	8.06	**7.34**
sofa	11.94	11.59	11.58	13.69	11.89	10.65	**10.62**	10.92	10.63
table	9.65	9.64	8.59	11.43	8.71	7.87	8.44	7.74	**7.53**
vessel	9.84	9.83	9.64	11.19	8.48	7.95	8.05	7.66	**7.46**
Average	10.00	10.19	9.59	11.92	9.97	8.46	8.82	8.54	**8.25**

Table 2. Comparisons on missing parts with CD multiplied by 10^3.

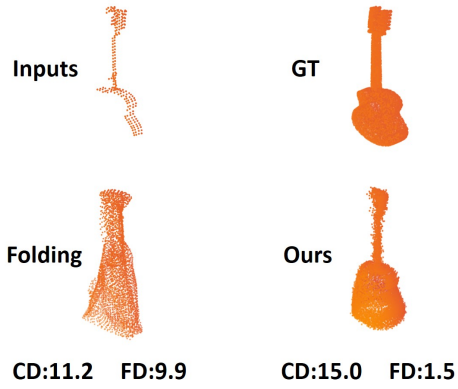

Figure 9. Comparisons of CD and FD multiplied by 10^3.

distance from each point in the input to its nearest neighbor in the output, which can be shown as follows:

$$FD(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2. \qquad (13)$$

As there is no complete ground truth model for KITTI, FD and Minimal Matching Distance (MMD) are used to evaluate the completion performance. MMD is defined as CD between the output and the car point cloud from ShapeNet which is closest to the output point cloud in terms of CD. This measures how much the output resembles a typical car.

4.2. Comparison on Completion Result

Experiments on ShapeNet. We qualitatively and quantitatively compare our RFNet on ShapeNet with several state-of-the-art point cloud completion methods. The test data is divided into two parts: data containing 8 known categories models same as the train data, and data containing 8 novel categories models different from the train data.

The quantitative results conducted on known categories are shown in Table 1, while that on novel categories in Table 3. Our network achieves the best performance on both known and novel categories models. Though CANet also performs well on known categories of ShapeNet, it needs a mean shape prior feature from a pre-trained network for each category, which is a little tough because categories of models cannot always be known before completion. Besides, our network improves a lot on FD, which means our RFNet can make fewer distortions during the completion process and preserve the original shapes better than other methods. However, our method actually introduces more information from incomplete models than most other methods. To better evaluate the completion performances, we remove the nearest neighbors of points in incomplete models from complete ones to compare the performances on missing parts. The result is presented in Table 2. We can see that our method still has the best performance for the completion of missing parts.

To intuitively compare the completion results, we choose a few models from the test data to make the qualitative comparison. As shown in Fig. 8, FC, Folding, PCN and TopNet create a good global shape, while losing most details from the incomplete model. MSN, GRNet and CANet can preserve details to some extent, while they still suffer from obvious distortions. Though NSFA can keep details much better than the other methods, its local feature aggregation operations increase computational cost greatly. Besides, it may mistake some discontinuous regions as details and have difficulty completing models with large and concentrated missing parts. Our RFNet can preserve details with fewer distortions and clearer textures, which is also capable of low computational cost as illustrated in Sec. 4.6.

Experiments on KITTI. We evaluate our network with the car category of real-world scans from KITTI. Our network is trained on ShapeNet for only about 0.8M iterations (10 epochs) in this section, without any fine-tuning on other

	Method	FC [32]		Folding [32]		PCN [32]		TopNet [24]		MSN [16]		GRNet [28]		NSFA [33]		Ours	
	Metric	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD	CD	FD
Similiar	bus	9.82	7.87	10.58	8.14	9.46	6.41	9.31	9.93	11.60	5.40	11.50	4.92	9.24	3.86	**8.98**	**2.42**
	bed	21.23	13.54	19.08	13.32	21.63	10.63	20.38	15.37	24.10	6.27	22.42	5.97	**17.30**	4.84	19.20	**4.35**
	bookshelf	15.12	10.53	14.88	10.39	14.79	8.52	14.12	12.69	16.20	6.45	14.91	6.22	**12.63**	4.78	12.91	**3.79**
	bench	10.81	8.88	10.55	9.43	11.02	7.58	10.16	11.08	10.80	5.00	11.47	5.38	**9.76**	3.82	9.79	**2.84**
	Average	14.20	10.20	13.80	10.32	14.20	8.28	13.40	12.27	15.67	5.78	15.08	5.62	**12.23**	4.33	12.72	**3.35**
Dissimiliar	guitar	9.92	9.26	9.06	9.30	10.40	8.61	9.88	10.11	10.40	2.40	8.88	4.04	8.72	3.30	**7.59**	**1.89**
	motor	14.56	11.97	15.56	14.49	14.75	11.48	14.30	14.52	15.50	**4.39**	11.83	4.51	**10.56**	**3.80**	10.88	4.99
	skateboard	12.00	7.77	11.91	7.49	12.04	6.56	9.26	9.63	11.70	4.00	11.30	3.73	8.68	2.88	**8.66**	**1.48**
	pistol	14.97	13.86	13.13	12.96	14.23	10.70	12.86	15.42	14.20	2.87	13.27	3.55	11.03	3.54	**9.74**	**2.90**
	Average	12.90	10.72	12.40	11.06	12.90	9.34	11.50	12.42	13.95	3.42	11.32	6.79	9.75	3.38	**9.22**	**2.82**

Table 3. Quantitative Comparisons on novel categories of ShapeNet with state-of-the-art methods with the metrics multiplied by 10^3.

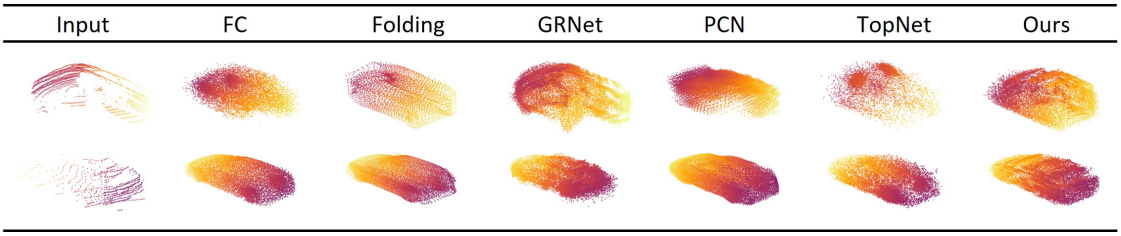

Figure 10. Qualitative comparisons on car category of KITTI. Our work can better capture and recover basic shapes of cars than the others.

Method	FC	Folding	PCN	TopNet	GRNet	Ours
FD	0.0331	0.0361	0.0308	0.0335	**0.0192**	0.0258
MMD	0.0148	0.0146	0.0158	0.0151	0.0374	**0.0146**
FD+MMD	0.0479	0.0507	0.0466	0.0486	0.0557	**0.0404**

Table 4. Quantitative comparisons on car category of KITTI.

datasets. The quantitative and qualitative results are shown in Table 4 and Fig. 10. The mean MMD of our RFNet is the smallest, which means our method has the best performance capturing the basic shape of cars and resembling typical cars. However, GRNet gets a small FD with a large MMD, which means GRNet does too much reconstruction, instead of completing input models. In order to make a trade-off between FD and MMD, we add them together to make an overall evaluation. Our network gets comparable results on FD but performs best on MMD and the overall evaluation.

4.3. What Does the Network Learn in the RFE?

Max pooling operation used in an Encode Cell or Recover Cell is actually a selection of key points which achieve the maximum value in feature dimensions. As shown in Fig. 11, we visualize key points selected by the Encode Cells and Recover Cell in the RFE. Points selected by the Encode Cells gradually move to the missing parts as the recurrent level increases, which proves that our network is capable of extracting features to complete the missing parts adaptively. Besides, key points selected by the Recover Cells are around those generated by the Encode Cells. It indicates the Recover Cell learns to aggregate information from inputs based on the output of Encode Cells.

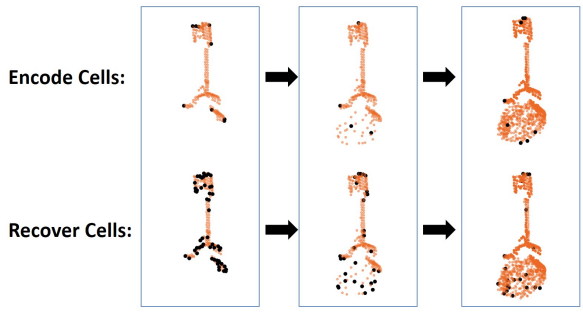

Figure 11. Key points captured by the Encode Cells and Recover Cells in the RFE. The black points denote key points selected.

4.4. Points Grown From Seeds in the FDC

In the FDC, the Initialize Cell generates an initial sparse model by the combination of the fully-connected network output and refined sampled points. We visualize the distributions of the points in output models grown from these two parts, respectively. As demonstrated in Fig. 12, points grown from sampled seeds form a contour for the original incomplete parts. Besides, points grown from the fully-connected network output fill up missing parts of models. These two parts are combined to make up the final output.

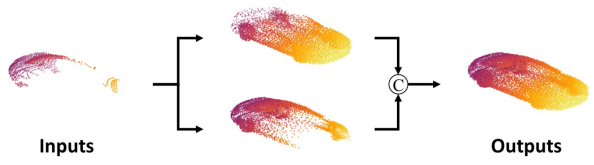

Figure 12. Points grown from seeds. The upper branch and lower branch denote points grown from generated seeds and sampled seeds, respectively.

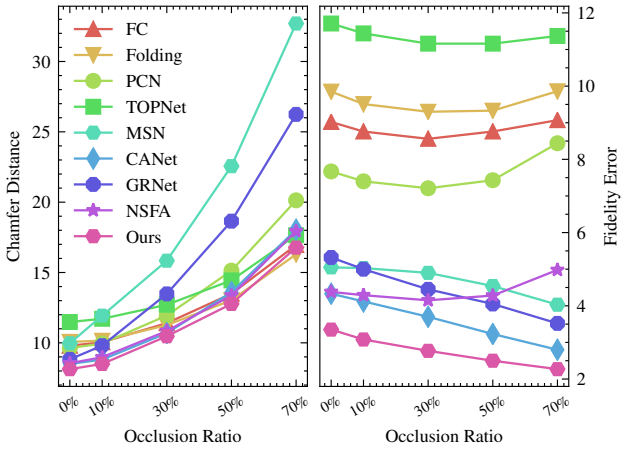

Figure 13. Quantitative comparison for occluded point clouds under the different occlusion ratios.

Method	Inference		Training requirements		
	Time(ms)	Memory(MB)	Batch	Iter (M)	Memory(GB)
PCN	6.68	973	32	0.3	11
TopNet	**5.09**	732	32	0.23	11
MSN	20.16	1417	160	0.23	8x11
GRNet	5.92	1719	32	1.09	2x11
CANet	9.22	973	32	0.27	11
NSFA	104.80	973	8	0.67	11
Ours	9.00	**710**	**32**	**0.23**	**11**

Table 5. Model efficiency comparison. Time and memory cost are evaluated on a Nvidia 2080ti GPU with a 2.9Ghz i5-9400 CPU.

4.5. Robustness for Occlusion

In real-world applications, point missing, as a common problem, can introduce extra noise to data and harm to completion. To further study the robustness of our method, we conduct experiments by occluding inputs with $\%p$ occlusion following the procedure in PCN [32] and CANet [26], demonstrated in Fig. 13. Our method performs best both in CD and FD, which confirms that our network is more robust against occlusion than former methods.

4.6. Comparison on Network Efficiency

In this section, we make a comparison on memory cost, time cost and training requirements. As presented in Table 5, our network also has comparable time cost and the lowest memory cost. Though GRNet is faster, it needs more than 2 times larger memory than ours. Besides, our work has a relatively low training requirement. Comparison on training requirements with several completion networks is also reported in Table 5. Note that our network only needs 11 GB to train. It will take about only about 0.08 M iterations to converge to a result better than former networks, 0.23 M iterations to the best result in 1 day, which is much faster than other methods.

4.7. Ablation Study

In this section, we evaluate the effect of different modules in our network, including the Encode Cells and the Recover Cells in the RFE, the Merge Layers and the Refine Cells in the RSP, the proposed Sampling Chamfer Distance(SCD) and Balanced Expansion Constraint (BEC). The experiments are conducted on known categories models of ShapeNet by removing the modules and retraining the network. We use CD and FD to evaluate the completion results, as illustrated in Table 6. Full network with all modules works the best. Removing any component decreases the performance, which indicates that each component contributes. Merge Layer and Refine Cell in the RSP contribute the most to reduce the completion error. Though other modules help relatively slighter, they can improve the final performance and accelerate convergence.

Enc	SCD	BEC	Rec	Ref	Mer	CD	FD	CD*
✓	-	-	-	-	-	9.35	6.86	9.69
✓	✓	-	-	-	-	9.27	6.58	9.81
✓	✓	-	-	✓	✓	8.44	3.58	8.94
✓	✓	✓	-	-	-	9.26	6.59	9.72
✓	✓	✓	✓	-	-	8.96	6.44	9.45
✓	✓	✓	✓	✓	-	8.80	6.34	9.29
✓	✓	✓	✓	✓	✓	8.13	3.35	8.46

Table 6. Quantitative comparison for the ablation study. Enc, SCD, BEC, Rec, Ref and Mer denote the Encode Cell, Sampling Chamfer Distance, Balanced Expansion Constraint, Recover Cell, Refine Cell and Merge Layer, respectively. CD* means CD measured at about 0.08M iterations to compare the convergence efficiency.

5. Conclusion

In this paper, we propose a novel point cloud completion network named RFNet, which is organized in multiple recurrent levels. Output model from the former level will be concatenated with the incomplete model and fed to the latter level as a "new incomplete model". RFNet is consist of three modules: RFE, FDC and RSP. The RFE extracts multiple global features for completion in different resolutions, and the FDC generates completed point clouds from coarse to fine. The RSP is used to introduce details from the original incomplete models to the outputs. Besides, we also propose a Sampling Chamfer Distance and a Balanced Expansion Constraint to better capture the shape differences and improve completion performances in the multilevel structure. Exhaustive experiments on ShapeNet and KITTI indicate that our RFNet can achieve state-of-the-art performances with less cost than former methods.

Acknowledges

We thank all authors, reviewers and the chair for the excellent contributions. This work is supported by the National Science Foundation 61836015.

References

[1] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[4] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast point r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9775–9784, 2019.

[5] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5868–5877, 2017.

[6] N Dinesh Reddy, Minh Vo, and Srinivasa G Narasimhan. Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1906–1915, 2018.

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

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[9] Xiaoguang Han, Zhen Li, Haibin Huang, Evangelos Kalogerakis, and Yizhou Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *Proceedings of the IEEE international conference on computer vision*, pages 85–93, 2017.

[10] Zhizhong Han, Xiyang Wang, Yu-Shen Liu, and Matthias Zwicker. Multi-angle point cloud-vae: unsupervised feature learning for 3d point clouds from multiple angles by joint self-reconstruction and half-to-half prediction. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10441–10450. IEEE, 2019.

[11] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 984–993, 2018.

[12] Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. Pf-net: Point fractal network for 3d point cloud completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7662–7670, 2020.

[13] Truc Le and Ye Duan. Pointgrid: A deep network for 3d shape understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9204–9214, 2018.

[14] Dongping Li, Tianjia Shao, Hongzhi Wu, and Kun Zhou. Shape completion from a single rgbd image. *IEEE transactions on visualization and computer graphics*, 23(7):1809–1822, 2016.

[15] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-gan: a point cloud upsampling adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7203–7212, 2019.

[16] Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and sampling network for dense point cloud completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11596–11603, 2020.

[17] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[18] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgbd data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018.

[19] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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*, pages 652–660, 2017.

[20] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[22] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019.

[23] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018.

[24] Lyne P Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 383–392, 2019.

[25] Chu Wang, Babak Samari, and Kaleem Siddiqi. Local spectral graph convolution for point set feature learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 52–66, 2018.

[26] Xiaogang Wang, Marcelo H Ang Jr, and Gim Hee Lee. Cascaded refinement network for point cloud completion. In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 790–799, 2020.

[27] Xin Wen, Tianyang Li, Zhizhong Han, and Yu-Shen Liu. Point cloud completion by skip-attention network with hierarchical folding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1939–1948, 2020.

[28] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. *arXiv preprint arXiv:2006.03761*, 2020.

[29] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4541–4550, 2019.

[30] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018.

[31] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018.

[32] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, pages 728–737. IEEE, 2018.

[33] Wenxiao Zhang, Qingan Yan, and Chunxia Xiao. Detail preserved point cloud completion via separated feature aggregation. *arXiv preprint arXiv:2007.02374*, 2020.