remote sensing

MDPI

Article

Learning Contours for Point Cloud Completion

Jiabo Xu [1], Zeyun Wan [2] and Jingbo Wei [2,*]

[1] School of Remote Sensing Information Engineering, Wuhan University, Wuhan 430072, China
[2] Institute of Space Science and Technology, Nanchang University, Nanchang 330031, China
* Correspondence: weijingbo@ncu.edu.cn

Abstract: The integrity of a point cloud frequently suffers from discontinuous material surfaces or coarse sensor resolutions. Existing methods focus on reconstructing the overall structure, but salient points or small irregular surfaces are difficult to be predicted. Toward this issue, we propose a new end-to-end neural network for point cloud completion. To avoid non-uniform point density, the regular voxel centers are selected as reference points. The encoder and decoder are designed with Patchify, transformers, and multilayer perceptrons. An implicit classifier is incorporated in the decoder to mark the valid voxels that are allowed for diffusion after removing vacant grids from completion. With newly designed loss function, the classifier is trained to learn the contours, which helps to identify the grids that are difficult to be judged for diffusion. The effectiveness of the proposed model is validated in the experiments on the indoor ShapeNet dataset, the outdoor KITTI dataset, and the airbone laser dataset by competing with state-of-the-art methods, which show that our method can predict more accurate point coordinates with rich details and uniform point distributions.

Keywords: point cloud; voxel; completion; attention

Citation: Xu, J.; Wan, Z.; Wei, J. Learning Contours for Point Cloud Completion. *Remote Sens.* **2023**, *15*, 4338. https://doi.org/10.3390/rs15174338

Academic Editors: Sisi Zlatanova, Jiju Poovvancheri, Dong Chen, Takis Mathiopoulos and Zhengxin Zhang

Received: 28 July 2023
Revised: 28 August 2023
Accepted: 30 August 2023
Published: 3 September 2023

1. Introduction

Remote sensing requires the construction of real-world dynamic scenarios. Different to optical cameras that capture two-dimensional (2D) images, the laser scanning with Light Detection And Ranging (LiDAR) devices acquire three-dimensional (3D) data for a better representation of spatial geometry. Three-dimensional (3D) structures are described typically in formats of depth image, mesh, point cloud, and voxel, where the point cloud is the most commonly used. However, due to the discontinuous material surfaces or coarse sensor resolutions, the generated point clouds may be locally sparse or missing, which is not conducive to further tasks such as semantic segmentation or classification [1]. Then, the shape completion is needed to estimate the geometry of objects by reconstructing complete point clouds from partial observations.

Traditional point cloud repair methods are based on the a priori structural characteristics, e.g., symmetry or semantics, to account for the missing points. These methods can only repair point clouds with a low missing ratio and obvious structural features. Presently, deep-neural-network-based methods have been the main stream for point cloud completion. The encoder–decoder framework is the most commonly used completion framework, as has been used in [2–9]. Some studies [10–14] used the generative adversarial network (GAN) to improve the training of the generator. Recently, the graph convolutional network (GCN) is introduced in some work [15,16] to address this topic. Fei et al. [17] investigated the latest and advanced algorithms for point cloud completion together with their methods and contributions. The related work will be introduced in the next section.

High-quality completion methods emphasize the integrity of details. A 3D shape contains faces, edges, and vertices, which have depth and so they occupy some volume. While all completion methods learn structure from the training set, the downsampled feature extraction modules commonly used in neural networks only capture the main

information. This leaves salient points or small irregular surfaces ignored and unable to be reconstructed. Wang et al. [18] proposed independent learning of edges to capture details. For many objects, such as chairs and computers, edges are sufficient to outline the whole structure. However, an object surface is not always flat, as the shape may resemble the sphere and torus. In this case, subtle shape details cannot be inferred from edges.

In addition to the overall structure, in this paper, we propose to learn contours for point cloud completion. Salient edges and irregular surfaces observed in the training data are expected to be reproduced by the proposed classifier. A new end-to-end neural network is designed for point cloud completion which follows the "point ⇒ voxels ⇒ points" framework in GRNet and VE-PCN, but the encoder and decoder are newly designed with transformers. The classifier is integrated into the decoder with novel loss functions to learn contours.

Although similar edge learning has been used in VE-PCN, our model is compact enough because it requires neither additional branches nor multiple independent down-sampling on the input for various grid scales. The contour learning is an improvement toward edge learning for richer a priori knowledge. The self-learning capability of the end-to-end structure makes our model an easier-to-use generator, with the potential to be further combined with more powerful frameworks such as generative adversarial networks and attention mechanisms.

The main contributions of the work are summarized.

1. We propose a new end-to-end network for point cloud completion with newly designed transformers for the encoder and decoder.
2. We propose a solution to learn object contours incorporating critical edges and irregular surfaces for point cloud completion.

The remainder of this article is arranged as following. In Section 2, related work on point cloud completion is reviewed. In Section 3, the proposed model structure is presented, and the contour learning is introduced. In Section 4, an experiment is performed on the ShapeNet dataset to demonstrate the performance of the proposed method by comparing with state-of-the-art completion methods. In Section 5, the in situ KITTI dataset is used to give a further evaluation of the proposed method. An airbone dataset is tested in Section 6 for remote sensing evaluation. The potential advantages and disadvantages of the proposed method are discussed in Section 7. Section 8 gives the conclusion.

2. Related Work

Deep-neural-network-based methods have been the main stream for point cloud completion to output key points of complete structures and generate denser point offsets based on the key points. The majority of the models are built with convolutional neural networks (CNNs). Achlioptas et al. [2] introduced the first encoder–decoder framework to implement the deep generation model for point clouds. Yang et al. [3] improved the auto-encoder model with a graph-based encoder and a two-consecutive-folding-based decoder (FoldingNet) to deform a canonical 2D grid onto a smoother 3D object surface. By combining the advantages of [2,3], Yuan et al. [4] proposed the point completion network (PCN) with fully connected networks to capture the overall shape of point clouds. PCN is further improved in [5] by folding multiple patches to generate continuous and smooth point cloud surfaces, and then combining these point cloud surfaces. Zhao et al. [6] suggested the 3D capsule networks as the auto-encoder, where the dynamic routing scheme and 2D latent space were deployed for restoration. To improve FoldingNet, Groueix et al. [7] proposed AtlasNet with K multilayer perceptrons to fold out K surfaces simultaneously. Tchapmi et al. [8] proposed the topology repesentation network (TopNet) with a multilayer tree structure as the decoder. Xie et al. [9] proposed the gridding residual network (GRNet) in which a gridding layer and a gridding loss were designed to regularize the disordered point cloud to 3D grids. Junshu et al. [19] proposed LAKe-Net by localizing aligned keypoints (LAKe) with a topology-aware keypoints-skeleton-shape prediction solution. Yingjie et al. [20] proposed a novel framework which learns a unified and structured latent space that encodes both

partial and complete point clouds. The codec framework shows good performance and excellent efficiency.

The generative adversarial network (GAN) is also adopted for point cloud completion. Zhang et al. [10] used GAN to reconstruct from a given partial input, which was pretrained on complete shapes to search for latent feature codes. Wen et al. [11] proposed the cycle transformations between the latent spaces of complete shapes and incomplete ones to build the bidirectional geometry correspondence. Miao et al. [12] designed a new encoder for neighboring point information in different orientations and scales as well as a decoder to output dense and uniform complete point clouds. Wen et al. [13] devised a dual-generators framework for point clouds generation, which implemented two GANs to learn effective point embeddings and refine the generated point clouds progressively. Xie et al. [14] presented the Style-based Point generator with Adversarial Rendering (SpareNet) by channel-attentive EdgeConv to fully exploit the local structures. Cheng et al. [1] proposed a GAN-based dense point cloud completion architecture with skip connections to a fully connected layer-based network for regenerating global feature. The processed point clouds can be successfully used for classification which shows the value of completion. GAN trains the generator in an alternative and adversarial way which is possible for finding better parameters.

The graph convolutional network (GCN) has also been used for the completion of point clouds. Pan [15] proposed an edge-aware GCN, in which edge features are captured against noise and then propagated with deep hierarchical graph convolution for completion. Shi et al. [16] proposed a graph-guided deformation network which treats the input data as controlling points and intermediate generation as supporting points, and it models the optimization guided by a GCN. These methods converts vertices into a graph for inference with deep learning.

Although the original shapes are partially missed, the above-mentioned studies output the full region point clouds. To alleviate the error caused by the change of original partial shapes, new methods were proposed to predict only the missing regions. Huang et al. [21] proposed the Point Fractal Network (PF-Net) with partial point clouds as the input, and only the missing point clouds are output instead of the whole object. It has a combined multilayer perception to extract multi-scale features and a point pyramid decoder to generate the missing point clouds hierarchically based on feature points. Wen et al. [22] formulated the prediction as a point cloud deformation process with a novel neural network to simulate an earth mover constraining that the total distance of point moving paths should be shortest.

The key or reference points are the basis to diffuse to more points, but their quality suffers from the adverse sparseness of the point cloud. An area is marked for diffusion when it has one or more points. As a result, the predicted reference points may be unreasonable or inaccurate in case of local sparsity. The homogeneous distribution of the reference points is not assured, which results in the non-uniform density of the diffused point cloud. Instead of predicting key points, GRNet [9] and VE-PCN [18] predict the key voxel grids to improve the completion quality. Voxel grids can divide the whole space into regular grids, which ensures that the reference points are not clustered together, thus solving the problem of non-uniform point density after diffusion.

Although the "point ⇒ voxels ⇒ points" framework is successfully used, GRNet and VE-PCN are not ideal enough for completion. VE-PCN is not an end-to-end network due to the independent branches in both the input and the encoder. On the input side, VE-PCN implements multi-scale sampling of point clouds with the help of cascaded farthest point sampling, which results in a high dimensionality of the network input. In fact, multi-scale sampling can be implemented and self-learned in the network, which can reduce unnecessary input. An important contribution of VE-PCN is an edge extraction to guide diffusion, but it depends on a point generator which is not learnable. As for GRNet, the transformation from a point cloud to voxels and the inverse transformation are much

too simple, which relies only on the eight convolutional layers for grids completion. GRNet also ignored the varying point importance as generated by the edge branch in VE-PCN.

3. Methodology

This section will detail the newly proposed model. The overall framework is composed of an encoder, a decoder, and a diffusion module (Figure 1). The codec framework is a general solution to point cloud processing. However, the new idea is that our encoder produces voxel features to indirectly extract point cloud features, which is guided by a classifier for diffusion. The proposed model is named as Point-Voxel-Point Network and abbreviated as PVP-Net for short.

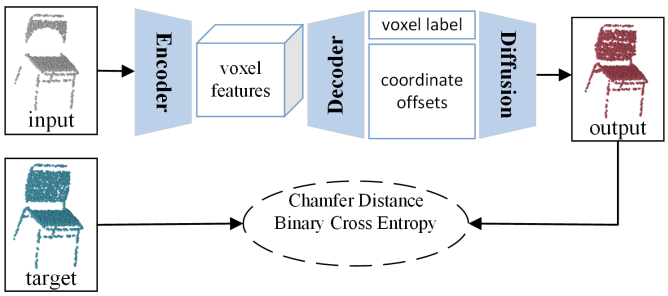


Figure 1. Framework of the method.

3.1. Encoder

The structure of the encoder is shown in the upper part of Figure 2. The point cloud coordinates are normalized to $[-1, 1]$ and input to the encoder. The encoder consists of a Patchify module and an Attention module. In the Patchify module, the point cloud is divided into non-overlapping grids, and their features are encoded separately. In our method, the feature of a grid is recorded as a vector of length 512. The Patchify operation consists of four steps. First, the space containing the complete point cloud is uniformly divided into $W \times W \times W$ grids. Second, the points within a grid are transformed to a local coordinate system with the grid center as the origin. Third, each local point coordinate is mapped to a feature by a multilayer preceptron (MLP). Lastly, all the features in a grid are aggregated into a vector by an aggregation function (e.g., AvgPool or MaxPool) to represent the patch feature. For an empty grid (without points in it), its patch feature is a learnable vector. This vector is shared by all empty grids and is determined during the optimization process. The Patchify process is shown in Figure 3.

Instead of the standard Patchify operation, a hierarchical Patchify strategy is suggested. The size of W in Patchify has a significant impact on the network. The decoder spans the space to $W \times W \times W$ grids. Since the transformer is used in the decoder, the interaction in transformer requires the square number of grids (W^6). The feature of a grid is coded in 512 words (2048 bytes). On the one hand, if the value of W is large, optimization may not be possible as the transformer is memory intensive. For example, $W = 32$ is a typical value which requires over 2048 gigabytes video card memory for GPU acceleration. On the other hand, a small W value may lose the scale details of the point cloud too quickly, resulting in the loss of geometric information. To solve this problem, a multi-stage strategy is proposed to gradually reduce the scale while keeping as much geometric information as possible.

As shown in Figure 4, the core of the hierarchical Patchify is grid subdivision and feature merging. It divides the grids into smaller grids for Patchify and then merges the features. In our method, a 2-stage hierarchical Patchify is used. For the 2-stage hierarchical Patchify process, the point cloud is pictured as a $1 \times 1 \times 1$ grid that is initially divided into coarse grids with the number $W/2 \times W/2 \times W/2$. Each coarse grid is further divided into eight fine grids, which undergo the Patchify to give a $n \times d$ matrix where n is the point number in a coarse grid and d is the feature length. In other words, each point is given a vector description of length d, and all the points in a fine grid share the same descriptor.

The $n \times d$ matrix is concatenated with the original coordinate matrix $n \times 3$ to fuse the feature of the coarse grids recorded as a $n \times (d + 3)$ matrix.

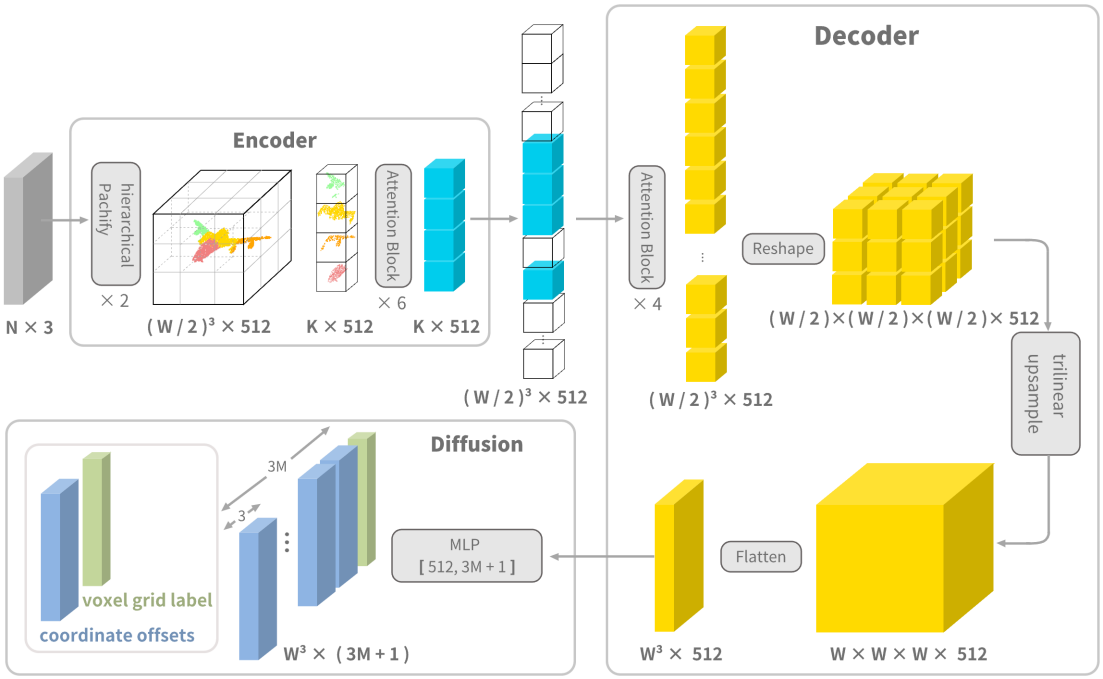


Figure 2. Architecture of the encoder and decoder. The space is divided into $W \times W \times W$ grids. N denotes the number of input points, K denotes the number of non-empty grids, and M denotes the number of point diffusion in a grid. "hierachical Patchify $\times$ 2" indicates that a two-stage hierachical Patchify is used. Six attention blocks are used in the encoder while four are used in the decoder. The vector length of a grid feature is 512.

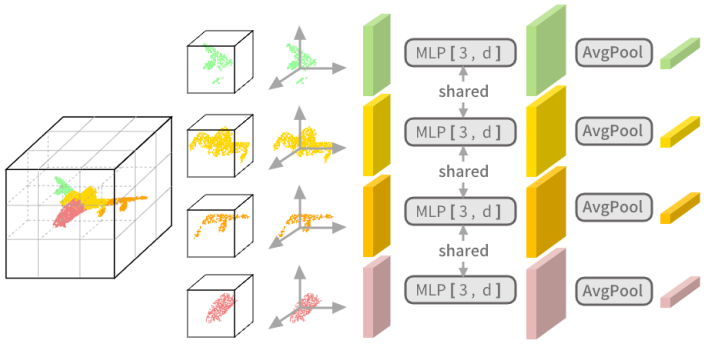


Figure 3. Patchify module. MLP (3, d) denotes the multilayer preceptron transforming the specific dimension from 3 to d, AvgPool denotes the average pooling, and d denotes the number of an expanded feature vector which is set to 512. The point cloud is divided into patches. Each patch is transformed to a local coordinate system, and its point coordinates are mapped through an MLP to length d. All mapped features are aggregated through an average pooling to obtain the feature of a patch.

The features extracted by hierarchical Patchify are fed to the attention module. Taking the strategy of masked autoencoders (MAE) [23], only features of non-empty grids are extracted. They are combined with position codes and then go through a series of transformer blocks for encoding. Since the encoder works only on non-empty grids, the amount of operations and speed of the operations are greatly reduced.

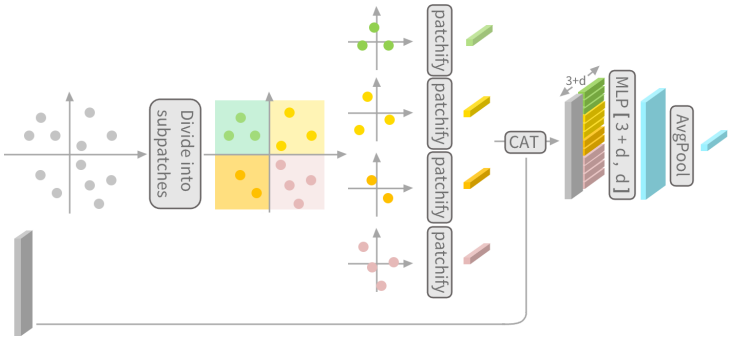


Figure 4. Two-stage hierarchical Patchify. MLP (3 + d, d) denotes the multilayer preceptron transforming the specific dimension from $3 + d$ to d, AvgPool denotes the average pooling, and CAT denotes the concatenation operation. A patch is divided into four smaller patches for Patchify, and the concatenated features are merged after MLP and AvgPool.

Specifically, let F consist of non-empty voxel features. $F \in R^{K \times d}$, where K is the number of non-empty voxels and $d = 512$. F is input to the attention blocks after summing with the position encodings. The 1D position embedding is used, which consists of W^3 of sinusoidal positional encoding computed according to [24]. There are two sub-layers in each attention block: a multi-headed attention layer enabling interaction between the input features, and a forward network. The multi-headed attention layer MHAttn is defined as

$$\mathrm{MHAttn}(Q, K, V) = (\mathrm{Head}_1 \oplus \ldots \oplus \mathrm{Head}_H) W^O, \tag{1}$$

$$\mathrm{Head}_h = \mathrm{Attn}(Q \cdot W_h^Q, K \cdot W_h^K, V \cdot W_h^V), \tag{2}$$

where $\oplus$ denotes the concatenation along the channel direction. $W_h^Q, W_h^K, W_h^V \in R^{d \times d_{\mathrm{head}}}$, and $W^O \in R^{H \cdot d_{\mathrm{head}} \times d}$ are learnable projection matrices. Each attention head employs a single-head dot product attention

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}(\frac{Q \cdot K^T}{\sqrt{d_{\mathrm{head}}}}) \cdot V \tag{3}$$

The parameters of the attention block are set to $H = 8$ and $d_{head} = 64$. Q, K, and V are all set to F; that is, F is mapped to $MHAttn(F, F, F)$ after each self-attention block. The forward network operates on each feature in the network separately. Following the implementation in [24], a two-layer feed-forward network is used with a rectified linear unit (ReLU) activation function after the first layer.

3.2. Decoder

The structure of the decoder is shown in the lower part of Figure 2. The input to the decoder is the features of all patches, including the empty patches of the empty grid and the non-empty patches of the encoder output. The decoded result is obtained by adding positional encoding to all the patch features and then passing through a series of attention blocks with the same configuration as the encoder and a trilinear upsampling layer.

3.3. Diffusion

The output of the decoder consists of two parts—a grid label and some coordinate offsets of the diffused points relative to the center of the voxel grid to which it belongs. The number of diffused points is preset as M that is linked to the density of predicted points. A typical value for M is 5 for the ShapeNet dataset [25]. Once again, it is noted that the reference points of our method are defined as the regular grid centers instead of the known or predicted points in a point cloud. The diffusion operation is illustrated in Figure 5. For each valid grid, diffusion is performed within the grid with M point

coordinates calculated using the offsets from the network output plus the coordinate of the grid center.

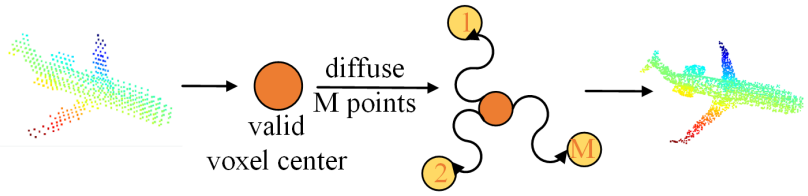


Figure 5. Diffusion for a valid voxel.

The grid labels generated by the decoder judge the validity of a grid. The label values are floating points which are mapped to the range [0, 1] through the sigmoid function. When the sigmoid value is greater than 0.5, the grid is valid and qualified for diffusion. In other words, invalid grids are not diffused because they may correspond to a vacant or extremely sparse space. Obviously, most grids are marked as invalid due to the sparsity of point clouds.

As far as the prediction regions are concerned, our method can perform completion both globally and locally. Completion is global when diffusion is performed within all valid voxels and no input points are kept. For local completion, all the input points are remained, and diffusion is performed within all valid voxels to ensure that the point number in each valid voxel is no less than M. Both methods have advantages and disadvantages. Local completion is preferred because our method is distinguished in judging the validation of voxels.

3.4. Training and Loss Functions

Complete data are required in training. The majority of the point coordinates are randomly extracted from a complete point cloud data and fed into the proposed network for prediction. There will be position deviation between the predicted point cloud and the true point cloud. In this case, the cyclic Chamfer distance is commonly used to evaluate the error between the point cloud coordinates obtained by diffusion and the ground truth, which is defined as $\mathcal{L}_{\mathrm{CD}}$ and calculated with

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

Here, S_1 and S_2 represent point clouds; N_{S_1} and N_{S_2} are the point numbers in these two sets; x and y are the coordinates of the point in each cloud; $\min(\cdot)$ represents the minimum distance between a point and a point cloud. Equation (4) searches for the average minimal distance in both directions.

The effectiveness of each grid is also concerned in our method but is not measured yet. To evaluate the impact of invalid grids, the binary cross-entropy is appended as an additional loss, which is defined as $\mathcal{L}_{\mathrm{BCE}}$ and calculated with

$$\mathcal{L}_{\mathrm{BCE}} = -\frac{1}{N} \sum_{i=1}^{N} [y_i \cdot \log(p(y_i)) + (1 - y_i) \cdot \log(1 - p(y_i))] \tag{5}$$

where N is the number of all labels, y_i represents the label for the i^{th} grid, and $p(y_i)$ represents the probability that y_i is valid. y_i is denoted as 1 or 0 to indicate that the i^{th} grid is valid or not, respectively. $p(y_i)$ is obtained using the sigmoid function on the first output of the last fully connected layer.

We observe that the importance of grids varies and can be classified into four categories, as shown in Figure 6. V_{existing} denotes the valid grids that exist in both input and target. V_{missing} denotes the valid grids missing in the input but present in the target. Analogous to the edges in VE-PCN, I_{contour} denotes the invalid grids that are adjacent to valid grids to construct a contour covering the surface of an object. The adjacent grids are the neighbored 26 grids in the $3 \times 3 \times 3$ space except for the center grid. I_{existing} denotes the invalid grids other than I_{contour}. Clearly, it is easy to identify V_{existing} and I_{existing} but difficult for V_{missing} and I_{contour} as they are the focus of completion. Therefore, the weights for these grids are different. The general classification error is defined as the confidence loss $\mathcal{L}_{\text{Confidence}}$ and calculated with

$$\begin{aligned} \mathcal{L}_{\text{Confidence}} = \alpha_1 L_{\text{BCE}}^{(V_{\text{exsiting}})} + \alpha_2 L_{\text{BCE}}^{(V_{\text{missing}})} \\ + \alpha_3 L_{\text{BCE}}^{(I_{\text{existing}})} + \alpha_4 L_{\text{BCE}}^{(I_{\text{contour}})}. \end{aligned} \tag{6}$$

The overall loss is the sum of the above two loss functions

$$\mathcal{L} = \mathcal{L}_{\text{CD}} + \mathcal{L}_{\text{Confidence}}. \tag{7}$$

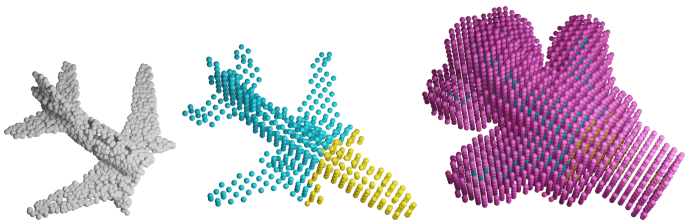


Figure 6. Grids are categorized for learning (blue for V_{existing}, yellow for V_{missing}, and purple for contour).

4. Experiment on ShapeNet

To validate the effectiveness of the proposed method, the first experiment was carried out on the ShapeNet dataset [25]. The results are compared with that of state-of-the-art models quantitatively and visually.

4.1. Experimental Scheme for ShapeNet

ShapeNet has 16 types of objects and a total of 15,011 samples. Among them, the number of samples in the training set is 12,137, and the number of samples in the test set is 2874. To assess the performance of our method, some state-of-the-art point cloud completion algorithms, including PCN [4], PCN-FC [4], 3D-Capsule [6], PF-Net [21], CDA [26], TopNet [8], CRN [27], GRNet [9], DPC [28], MSN [5], VE-PCN [18], and LAKe-Net [19] are tested on the ShapeNet dataset.

As far as the newly proposed method is concerned, parameters for training and test are given. Before training, the point cloud coordinates in the dataset were normalized to the range $[-1, 1]$. W is the grid quantity at a single direction which is set to 26 to suit for our hardware resources. The number of allowed diffusion points M is set to 5 for valid voxel grids which brings ignorable impact to the accuracy assessment. The weights for the confidence loss in Equation (6) are 2, 15, 0.3, and 2, respectively. To train the newly proposed network, the Adam optimizer was used to optimize the network for 100 epoches. The initial learning rate of the optimizer is 0.001, which is reduced to half after every 20 rounds until it reaches the minimum value of 0.00001. These parameters are shared by the following tests on two diverse datasets except for the training epoches which are 250 and 200, respectively.

4.2. Experimental Results of ShapeNet

The Chamfer distance in Equation (4) is used for evaluation. Scores of the competing algorithms are mostly extracted from the results in [18] except for PCN and three values of PF-Net, which were tested with our code. The latter is tested with our code because PF-Net in [21] was tested for only 13 classes. Data were assessed assuming that all the input point clouds were normalized to $[-0.5, 0.5]$, and the scores are magnified 10,000 times to observe the deviations to ground truth.

The normalized and scaled Chamfer distances were measured and the results are demonstrated in Table 1, where the best scores are marked in bold. The evaluation shows that our algorithm is better than all the competing algorithms for 13 classes among the total 16 classes, including Airplane, Bag, Cap, Car, Chair, Guitar, Knife, Lamp, Motorbike, Pistol, Rocket, Skateboard, and Table. On average, VE-PCN and our method outperform other competing algorithms significantly, while our method achieves 19% lower error than VE-PCN.

Table 1. Chamfer distance of overall point cloud completion on the ShapeNet dataset ($[-0.5, 0.5]$ normalized and 10,000 magnified; the best are in bold).

	PCN	PCN-FC	3D-Capsule	PF-Net	CDA	TopNet	CRN	GRNet	DPC	MSN	VE-PCN	LAKe-Net	Ours	Ours (No Contour)
Airplane	26.418	28.518	18.285	2.505	33.550	17.675	11.047	4.526	10.301	9.906	3.382	2.916	**1.164**	3.969
Bag	11.609	11.722	15.515	8.490	15.854	13.571	6.253	6.048	4.543	6.118	3.466	4.607	**3.211**	4.217
Cap	9.451	11.089	16.590	11.975	14.473	13.191	6.709	6.334	3.218	4.918	3.098	5.218	**2.859**	3.628
Car	7.254	7.667	8.288	5.115	8.344	8.419	4.776	5.704	3.714	8.214	3.480	3.927	**3.397**	4.165
Chair	4.675	5.165	4.576	4.570	6.446	6.026	3.116	4.286	2.811	2.666	2.476	2.536	**2.420**	3.673
Earphone	10.821	10.437	11.119	6.865	14.026	11.921	6.117	5.321	4.991	8.309	**3.239**	3.604	3.497	3.384
Guitar	0.996	1.134	1.361	0.995	1.289	1.403	0.690	1.309	0.869	0.641	0.750	0.687	**0.522**	0.768
Knife	2.998	3.110	5.075	2.275	5.235	3.495	2.015	1.700	3.518	2.604	1.603	1.249	**0.623**	1.374
Lamp	9.421	10.592	10.802	8.385	12.225	10.619	5.652	4.443	5.809	5.618	3.226	4.467	**2.074**	3.316
Laptop	3.236	3.464	3.102	2.730	3.653	3.941	2.534	3.745	1.854	**1.639**	2.197	1.868	2.279	2.184
Motorbike	6.005	6.398	7.425	4.555	6.706	7.033	3.434	4.237	4.081	4.692	2.717	3.782	**2.684**	3.385
Mug	9.522	10.788	11.703	7.420	11.877	11.706	6.731	8.022	6.802	**5.407**	4.938	5.427	5.823	3.976
Pistol	12.168	11.131	10.245	2.480	27.701	11.038	6.428	3.482	6.505	5.524	4.884	5.386	**1.633**	4.772
Rocket	10.035	10.853	9.058	2.140	16.029	8.075	4.750	1.999	5.378	4.992	1.574	3.629	**1.086**	2.084
Skateboard	3.504	3.858	3.458	1.985	4.464	4.321	2.288	2.835	1.804	2.431	1.780	1.923	**1.356**	2.183
Table	5.990	6.789	6.463	4.645	8.010	6.973	3.805	4.325	3.946	3.199	2.798	3.184	**2.479**	3.426
Average	8.381	8.920	8.941	4.821	11.868	8.713	4.772	4.270	4.384	4.805	2.851	3.401	**2.319**	3.156

Figures 7–9 show the rendered point clouds demonstrated in 2D images. In these figures, the rough structures can be built by all algorithms, but the global prediction algorithms (PCN and 3D-Capsule) fail to reconstruct details. The local prediction algorithms (PF-Net and ours) preserve far richer details than PCN and 3D-capsule. By comparing our results to PF-Net's, it is confirmed that the predicted points of our method are distributed in a more uniform way, which is friendly of visual understanding. This can be explained with the basis points in our method that are from regular voxel grids.

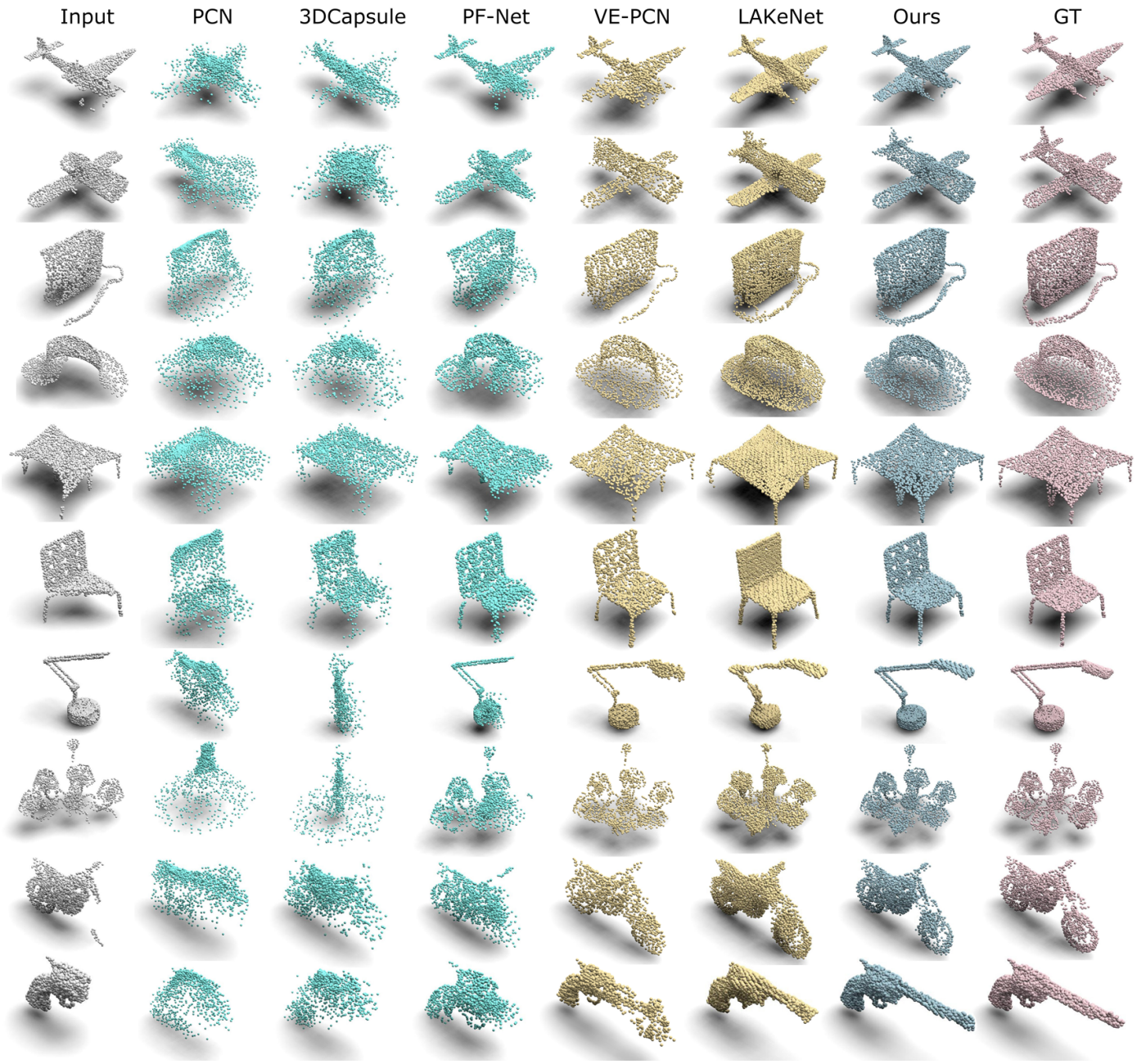


Figure 7. Visualization of the predicted point clouds.

Figure 8. *Cont.*

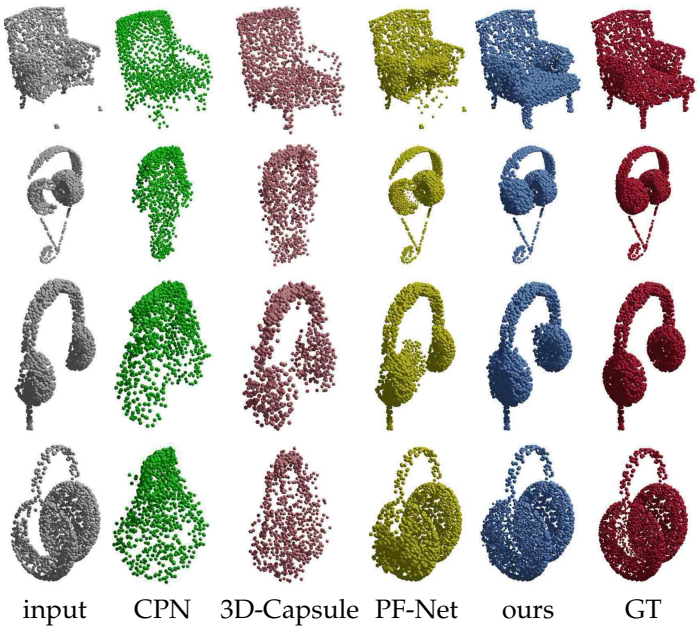


Figure 8. Visualization of the predicted point clouds for chairs and earphones.

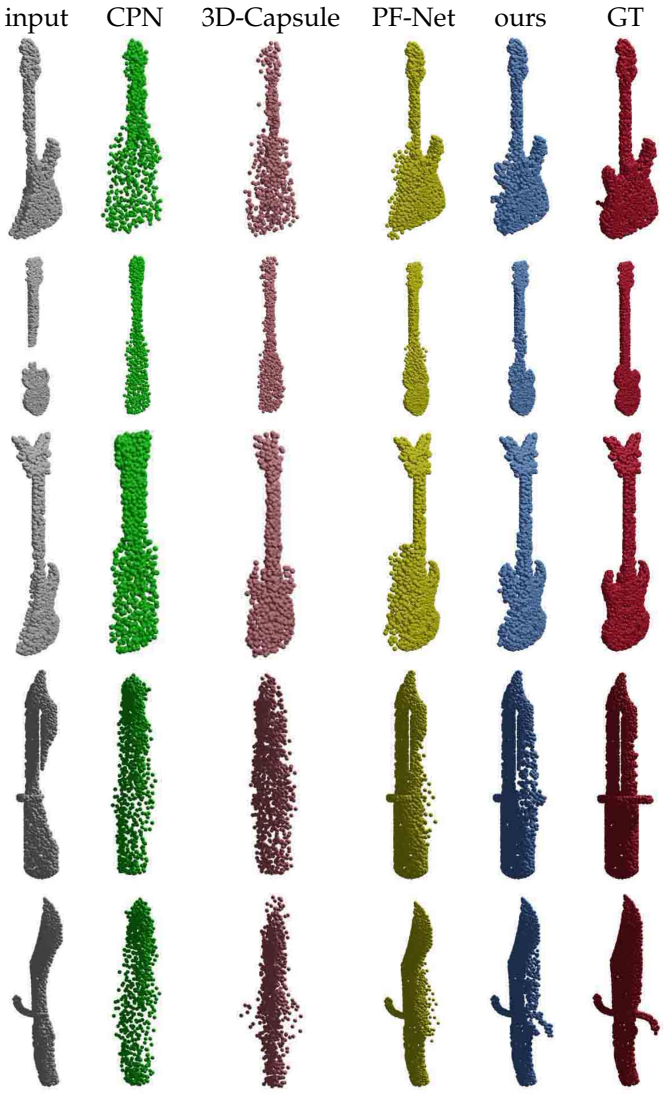


Figure 9. *Cont.*

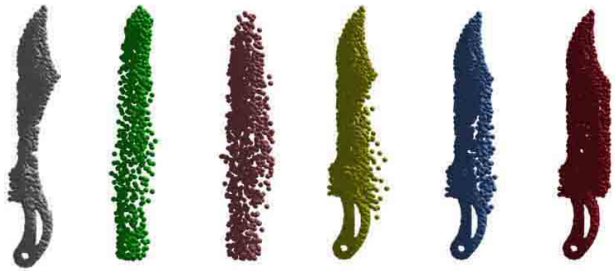


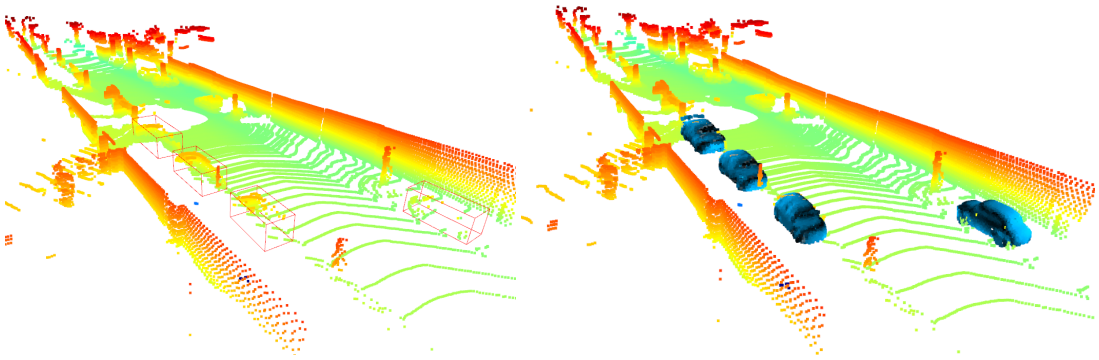
Figure 9. Visualization of the predicted point clouds for guitars and knives.

5. Experiment on KITTI

The second experiment is carried out on the KITTI dataset [29] for benchmark. The results are compared with that of state-of-the-art models quantitatively and visually.

5.1. Experimental Scheme for KITTI

KITTI is an in situ dataset consisting of a sequence of real-world Velodyne LiDAR scans. In each frame, the car objects are extracted according to the 3D bounding boxes (Figure 10a), which results in 2401 partially observed samples. The partial point clouds in KITTI are highly sparse without complete point clouds as ground truth. Cars in the KITTI dataset are targeted for reconstruction. Since the dataset has no complete point clouds for training, the a priori car models are learned from the ShapeNet dataset where the complete cars are used. Specifically, for each car in KITTI, the car sample in ShapeNet that has the minimum Chamfer distance to it is selected for training.



(**a**) (**b**)

Figure 10. Original and predicted frame for KITTI. (**a**) KITTI frame. (**b**) Completion result with the proposed method.

The results of some state-of-the-art point cloud completion algorithms, including 3D-Capsule [6], PCN [4], PF-Net [21], AtlasNet [7], FoldingNet [3], TopNet [8], MSN [5], GRNet [9], and SparNet [14] are also compared. It is noted that the outcomes of 3D-Capsule, PCN, and PF-Net are from our codes, while the results of AtlasNet, FoldingNet, TopNet, MSN, GRNet, and SparNet are cited from the original papers. Some algorithms used for the ShapeNet test in Section 4 are not included due to the lack of their experimental results. The parameters for the proposed method were the same as those used in the ShapeNet test except that the network was trained 250 epoches for the KITTI dataset.

5.2. Metrics

The point clouds in KITTI are gathered in situ without ground truth for reference, which is not applicable for the standard Chamfer distance as is used in the ShapeNet dataset. In this case, three new metrics are involved for evaluation, which are temporal consistency, fidelity, and the minimum matching distance. The first metric measures the

differences between frames. The last two metrics measure the output deviations from the input and training objects, respectively. Before the assessment, all the input point clouds were normalized to $[-1, 1]$. The scores measured with the following metrics were magnified 1000 times and demonstrated in Table 2.

Table 2. Evaluation on the results on the KITTI dataset (the best are in bold).

	3D Capsule	PCN	PF-Net	AtlasNet	FoldingNet	TopNet	MSN	GRNet	SparNet	Proposed
Temporal Consistency	0.836	1.557	0.374	0.700	1.053	0.568	1.951	0.313	**0.249**	0.273
Fidelity	1.527	2.235	**0**	1.759	7.467	5.354	0.434	0.816	1.461	**0**
Minimum Matching	1.218	1.366	0.613	2.108	0.537	0.636	2.259	0.568	0.368	**0.319**

5.2.1. Temporal Consistency

The Chamfer distance between the consecutive frames is measured with the temporal consistency. The metric is calculated frame by frame and then summarized to give a general assessment.

Suppose that the i^{th} frame and the $(i+1)^{th}$ frame are two consecutive frames in KITTI with the quantity of N_i shared cars in each frame, and car_k^i and car_k^{i+1} represent the completion results of the k^{th} car in the two frames, respectively; the temporal consistency for the i^{th} frame is abbreviated as C_i and calculated with

$$C_i = \frac{1}{N_i} \sum_{k=1}^{N_i} \mathrm{CD}(\mathrm{car}_k^i, \mathrm{car}_k^{i+1}) \tag{8}$$

where $\mathrm{CD}(\cdot)$ denotes the Chamfer distance between two collectives.

If the number of consecutive pairs of frames is K, then the overall consistency is calculated with

$$\mathrm{Consistensy} = \frac{1}{K} \sum_{k=1}^{K} C_k \tag{9}$$

5.2.2. Fidelity

Fidelity is measured with the single-directional Chamfer distance to evaluate how the input structures are preserved by the output. Supposed that the total number of cars in KITTI is N_{cars}, car_i denotes the i^{th} input car, and car_i' denotes the completion result for the i^{th} car, the fidelity is computed as

$$\mathrm{Fidelity} = \frac{1}{N_{\mathrm{cars}}} \sum_{i=1}^{N_{\mathrm{cars}}} \mathrm{CD}(\mathrm{car}_i, \mathrm{car}_i') \tag{10}$$

where $\mathrm{CD}(\cdot)$ denotes the Chamfer distance between two collectives.

5.2.3. Minimum Matching Distance

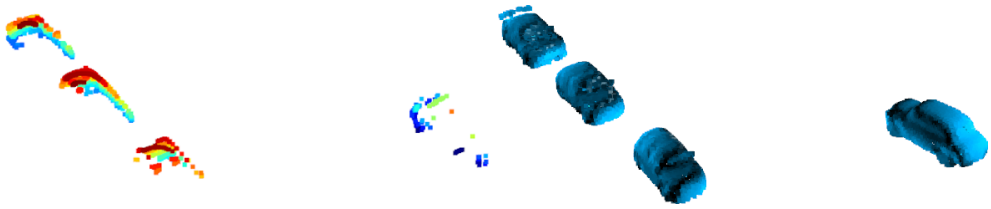
Since the a priori car models are trained from the ShapeNet dataset, the Chamfer distance between the predicted output for KITTI and the original car point clouds in Shapetnet can be evaluated. It describes the similarity between the training sets and the prediction sets.

5.3. Experimental Results on the KITTI Dataset

The prediction results by the proposed method are partly demonstrated in Figures 10–12 for visual identification. The results of competing algorithms are not presented because they are too similar to be distinguished if observed from far viewpoints. The reconstructed point sets show that car structures can be clearly reconstructed.

When the digital evaluations are concerned, Table 2 shows that our method achieves the best scores on fidelity and minimum matching distance, and it ranks in the top group with regard to temporal consistency. The fidelity score is 0 for PF-Net and our method

because the two algorithms keep all input points with only the missing components for prediction.



(**a**) (**b**)

Figure 11. Original and predicted cars for KITTI. (**a**) Cars in KITTI. (**b**) Cars predicted with the proposed method.

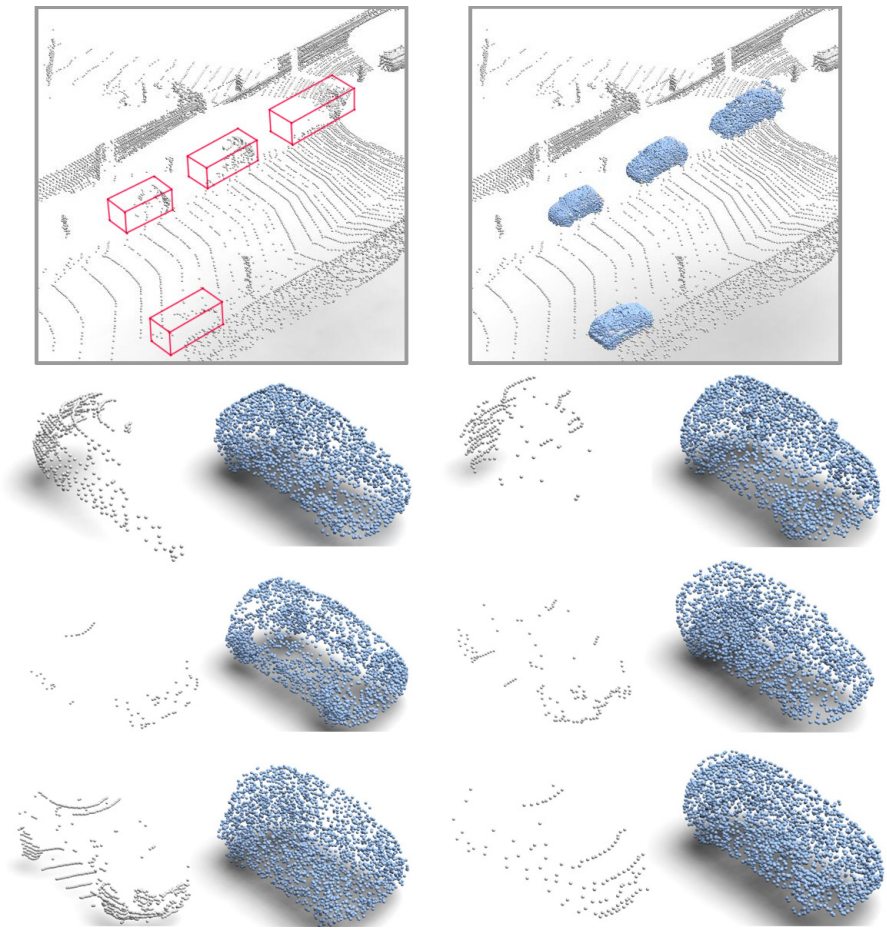


Figure 12. Visualization of the predicted point clouds for the KITTI dataset.

6. Experiment on the ALS Dataset

Ground completion is a key task. The 3D scenes obtained by matching satellite images usually do not contain ground structures, which is not applicable to some applications monitoring land cover changes, such as the height estimation of crops or forests. If the ground structure can be predicted using a point cloud completion model, canopy height can be obtained from the difference between vegetation and ground. This can lead to finer biomass assessment. To this end, an experiment is performed for remote sensing purposes.

The airborne laser scanning (ALS) dataset is used in the experiment, which contains three point cloud scenarios: dense vegetation plus gentle slope, dense vegetation plus steep slope, and sparse vegetation plus steep slope. The point clouds in the dataset are labeled with ground points and non-ground points. As for the scale of training and test,

the first scene contains 794 training samples and 88 test samples, the second scene contains 1625 training samples and 180 test samples, and the last scene contains 294 training samples and 32 test samples, respectively. The total training samples are 2713 and the total test samples are 300. In the training stage, ground points were removed from the scenes as the input, whereas the output of the model presented the missing ground structure. In the sparse + steep test, each point cloud sample contains roughly 12,000 points, of which the number of ground points is about 3000, with a missing percentage of about 25%. In the dense + slow and dense + steep tests, each point cloud sample contains roughly 18,000 points, of which the number of ground points is about 4000, for a missing proportion of about 22%.

Our method is compared with PCN, PF-Net, VE-PCN, PS-Net, and LAKe-Net. The parameters for the proposed method were the same as those used in the ShapeNet test, except that the network was trained 200 epoches for the ALS dataset. The experimental results are shown in Figures 13–15. The evaluation in Table 3 shows clearly that our method can achieve the best completion results compared with competing methods.

Table 3. Chamfer distance assessment on the ALS dataset (10,000 times magnified and the best are in bold).

	PCN	PF-Net	VE-PCN	PS-Net	LAKe-Net	Ours
dense + slow	18.54	14.28	14.20	15.65	13.53	**13.27**
dense + steep	31.23	22.25	21.35	22.36	20.91	**20.06**
sparse + steep	19.35	17.45	16.98	17.25	17.36	**16.90**

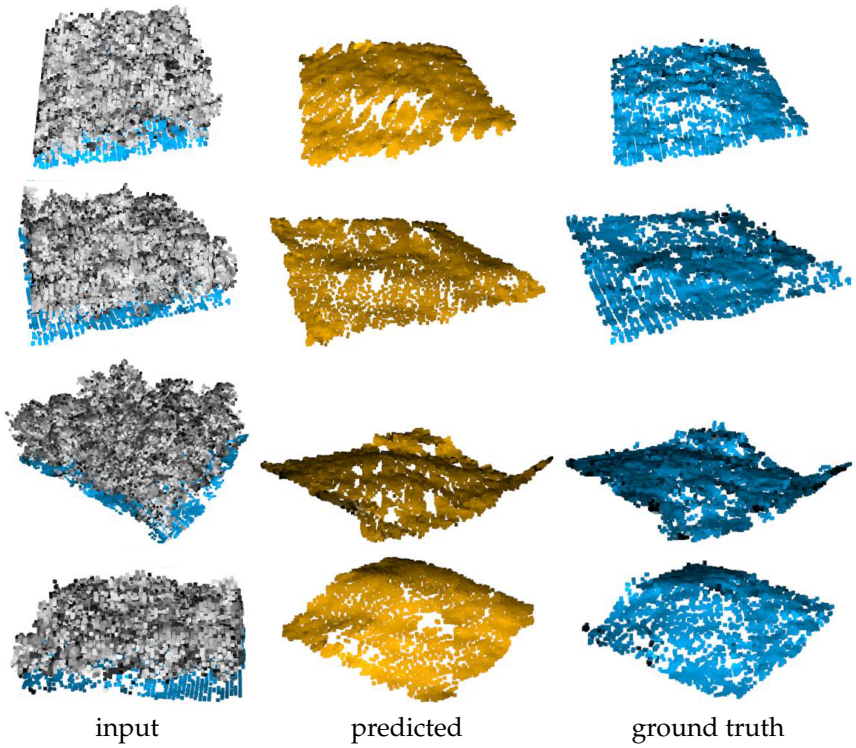


Figure 13. Visualization of the predicted point clouds for the dense+slow type of the ALS dataset. Each column represents an object.

Figure 14. *Cont.*

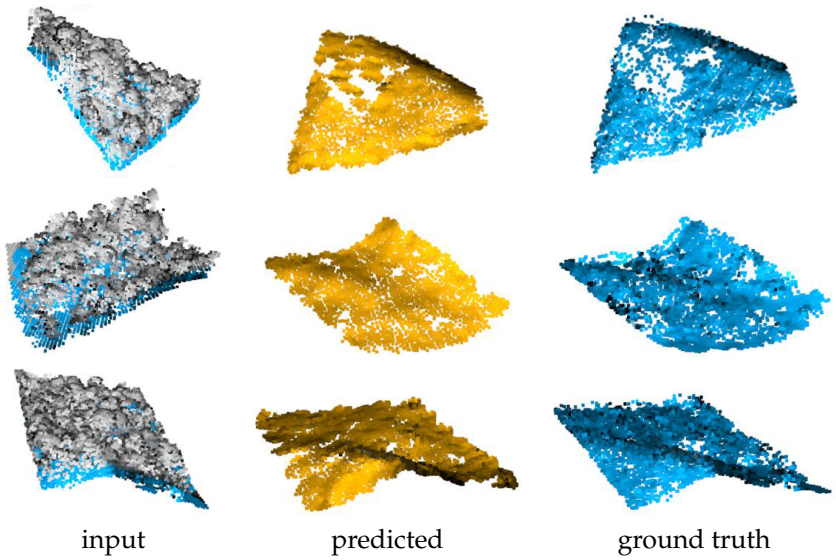


Figure 14. Visualization of the predicted point clouds for the dense + steep type of the ALS dataset.

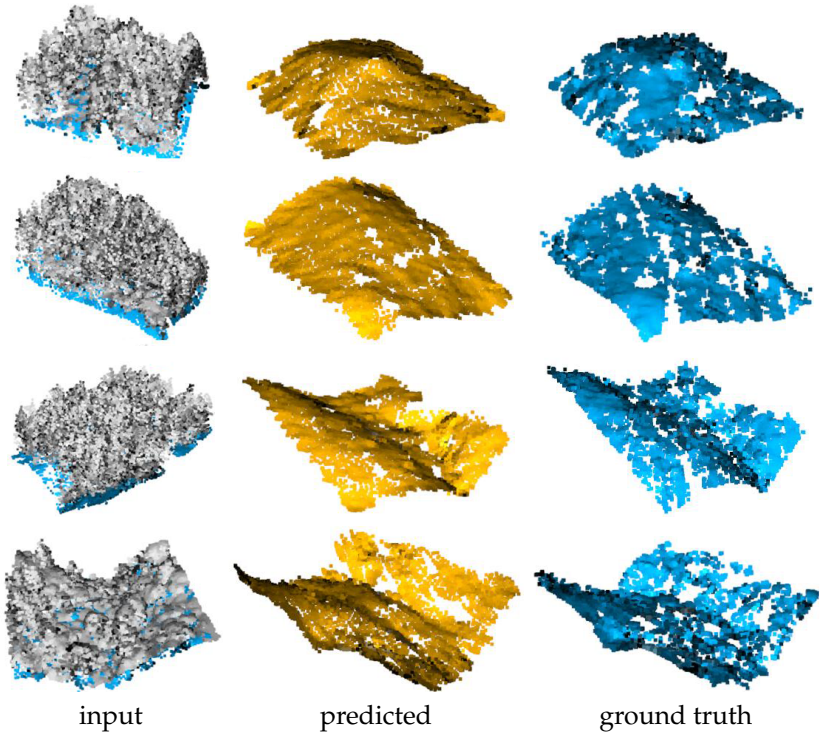


Figure 15. Visualization of the predicted point clouds for the sparse + steep type of the ALS dataset.

7. Discussion

Attention and contour learning are newly designed in our model. The necessity of the two models is uncovered in this section. In addition, the point cloud completion methods can be divided into two categories, namely overall prediction and missing prediction. Both types have advantages and disadvantages. Both methods can be used for the proposed method, but the missing prediction is preferred in the experiment. The detail loss and structural integrity are observed to compare the difference of the two methods.

7.1. Ablation Study

An ablation study was conducted to verify the effectiveness of the model structure and the newly proposed loss function. In the study, the attention blocks in the encoder and decoder are replaced with the U-Net structure based on 3DCNN. The effect of using

and not using contour loss is compared. The modified model was trained and tested on ShapeNet. The experimental results are shown in Table 4 and assessed with the Chamfer distance. The results show that the best results can be achieved by using both attention blocks and contour loss, which indicates that both the model and the loss function are effective and critical for performance improvement.

Table 4. Chamfer distance assessment on the ShapeNet dataset for ablation study (10,000 times magnified).

3DCNN	3DCNN + Contour	Attention	Attention + Contour
5.616	2.402	3.156	2.319

7.2. Impact of Contour Learning

In the newly proposed cost function, the importance of the grid is assigned to be variable. Its necessity can be presented by a confidence map. In order to plot the heatmap under the top view as shown in Figure 16, the middle layer of the voxel grid label predicted by the network is taken out. The results show that when the contour loss is not used in training (by setting $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 1$), the classification results of the contours are of low confidence. When the contour loss is used, the confidence of the classification is significantly improved, and a clearer contour can be seen from the confidence map.

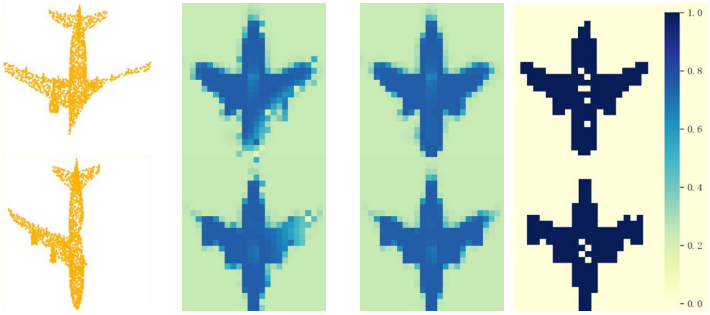


Figure 16. Confidence map with or without the contour learning.

7.3. Impact of Grid Size

The grid size is given by the important parameter W for tradeoff. A larger value of W allows for a higher voxel resolution, thereby preserving more shape details. However, it also increases the computational burden. On the other hand, an extraordinary large value of W implies capturing detailed contours, which brings challenges on the complexity of the network and the amount of training data. From the perspective of numerical evaluation, there exists an optimal W that matches the density of the points. In other words, a larger value of W is not necessarily better nor is a smaller value. The optimal W is linked to the point density.

In the experiment, W was set to 26 for all the tests. This value is the maximum value that can be supported by the computational resource used in this study (NVIDIA 2060 with 6GB memory). The training spends around 3 days for the ShapeNet dataset and 1 day for the KITTI dataset and the ALS dataset. Since the point cloud data have been normalized to the range of $[-1, 1]$ before being inputted to the network, the grid size is $2/26$.

7.4. Impact of Decision Threshold for Contour Learning

When the sigmoid function is used for normalization, the decision threshold for a typical binary classifier is commonly set to 0.5. However, Esposito et al. [30] found that 0.5 is suboptimal for imbalanced data. Point cloud completion also faces the challenge of data imbalance, so 0.5 may not be the ideal threshold. To find the truth, an experiment was added by changing the threshold within the range of [0.3, 0.7] and comparing the results

on ShapeNet, as shown in Figure 17. The results indicate that setting the threshold to 0.5 is appropriate. The difference between the worst and best results is only 0.087.

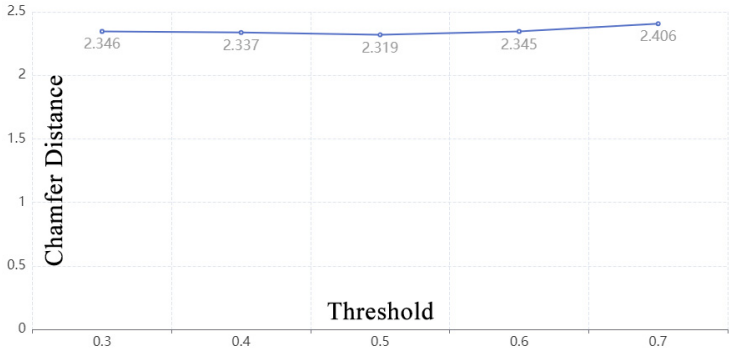


Figure 17. Chamfer distance on ShapeNet with varied decision threshold.

7.5. Detail Loss

In terms of details in the prediction results, the missing prediction methods tend to produce more details than the overall prediction methods. Missing prediction transfers the full details of the input clouds to the output part. Overall prediction, however, takes no additional constraint on the input, which is easy to lose details when the network reconstruction quality is not satisfied enough. Taking the car completion results in Figure 18 as an example, the completion results of the overall prediction methods fail to reconstruct the detail of the front windshield, while the completion methods of missing prediction can retain it.

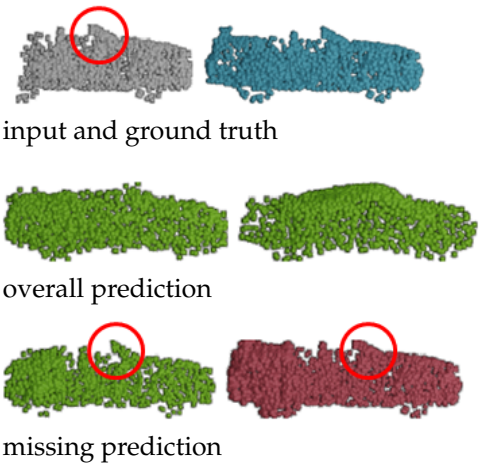


Figure 18. Detailed comparison between overall prediction and missing prediction. Points in red circles show significant difference under diverse strategies.

7.6. Structural Integrity

From the perspective of surface smoothness, the overall prediction results have smoother surfaces, while the missing prediction results may have gaps between the input and output. Figure 19 presents an example for integrity check. Despite the loss of detail or uneven density in the overall prediction, the final result is visually complete.

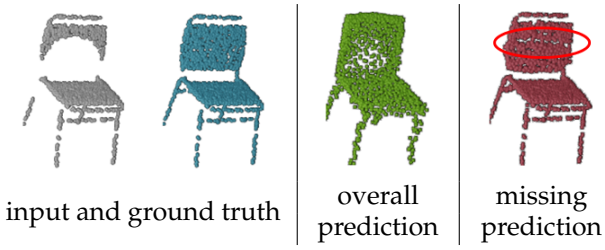


Figure 19. Integrity comparison between overall prediction and missing prediction. Missing prediction shows better consistency in the red circle area.

8. Conclusions

A learning-based model is proposed for precise and high-fidelity point cloud completion. In the method, the regular voxel points are selected as reference points to ensure uniform point density, and an end-to-end neural network is constructed for the cyclic conversion framework between a point cloud and voxels. A classifier is trained to remove the vacant or sparse grids and keep the valid voxels for diffusion. The deliberately designed loss function makes the classifier learn the prior knowledge of object contours.

The proposed method is named Point-Voxel-Point Network, or PVP-Net, tested on the ShapeNet dataset, and compared with 11 state-of-the-art algorithms. The Chamfer distance evaluation confirms that the new method outweighs all competing methods by 14% or higher. The visual comparison shows that our method can produce uniformly distributed points with higher details. Additional experiments on the KITTI dataset and the airbone laser dataset confirm the effectiveness of the proposed method for the in situ completion.

Author Contributions: Data curation, J.X. and Z.W.; Investigation, J.X. and J.W.; Software, J.X. and Z.W.; Writing—original draft, Z.W.; Writing—review and editing, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (No. 42267070).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cheng, M.; Li, G.; Chen, Y.; Chen, J.; Wang, C.; Li, J. Dense Point Cloud Completion Based on Generative Adversarial Network. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5701310. [CrossRef]
2. Achlioptas, P.; Diamanti, O.; Mitliagkas, I.; Guibas, L. Learning Representations and Generative Models for 3D Point Clouds. In Proceedings of the 35th International Conference on Machine Learning (ICML), Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 40–49.
3. Yang, Y.; Feng, C.; Shen, Y.; Tian, D. FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation. In Proceedings of the 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 206–215.
4. Yuan, W.; Khot, T.; Held, D.; Mertz, C.; Hebert, M. PCN: Point Completion Network. In Proceedings of the 6th International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018; pp. 728–737.
5. Liu, M.; Sheng, L.; Yang, S.; Shao, J.; Hu, S.M.; Assoc Advancement Artificial, I. Morphing and Sampling Network for Dense Point Cloud Completion. In Proceedings of the 34th AAAI Conference on Artificial Intelligence , New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11596–11603.
6. Zhao, Y.; Birdal, T.; Deng, H.; Tombari, F.; Soc, I.C. 3D Point Capsule Networks. In Proceedings of the 32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 1009–1018.
7. Groueix, T.; Fisher, M.; Kim, V.G.; Russell, B.C.; Aubry, M. A Papier-Mache Approach to Learning 3D Surface Generation. In Proceedings of the 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 216–224.
8. Tchapmi, L.P.; Kosaraju, V.; Rezatofighi, S.H.; Reid, I.; Savarese, S.; Soc, I.C. TopNet: Structural Point Cloud Decoder. In Proceedings of the 32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 383–392.
9. Xie, H.; Yao, H.; Zhou, S.; Mao, J.; Zhang, S.; Sun, W. GRNet: Gridding Residual Network for Dense Point Cloud Completion. In *Computer Vision—ECCV*; Springer International Publishing: Berlin/Heidelberg, Germany, 2020; pp. 365–381.

10. Zhang, J.; Chen, X.; Cai, Z.; Pan, L.; Zhao, H.; Yi, S.; Yeo, C.K.; Dai, B.; Loy, C.C. Unsupervised 3D Shape Completion through GAN Inversion. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 1768–1777.
11. Wen, X.; Han, Z.; Cao, Y.P.; Wan, P.; Zheng, W.; Liu, Y.S. Cycle4Completion: Unpaired Point Cloud Completion using Cycle Transformation with Missing Region Coding. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 13075–13084.
12. Miao, Y.; Zhang, L.; Liu, J.; Wang, J.; Liu, F. An End-to-End Shape-Preserving Point Completion Network. *IEEE Comput. Graph. Appl.* **2021**, *41*, 20–33. [CrossRef] [PubMed]
13. Wen, C.; Yu, B.; Tao, D. Learning Progressive Point Embeddings for 3D Point Cloud Generation. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 10261–10270.
14. Xie, C.; Wang, C.; Zhang, B.; Yang, H.; Chen, D.; Wen, F. Style-based Point Generator with Adversarial Rendering for Point Cloud Completion. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 4617–4626.
15. Pan, L. ECG: Edge-aware Point Cloud Completion with Graph Convolution. *IEEE Robot. Autom. Lett.* **2020**, *5*, 4392–4398. [CrossRef]
16. Shi, J.; Xu, L.; Heng, L.; Shen, S. Graph-Guided Deformation for Point Cloud Completion. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7081–7088. [CrossRef]
17. Fei, B.; Yang, W.; Chen, W.M.; Li, Z.; Li, Y.; Ma, T.; Hu, X.; Ma, L. Comprehensive Review of Deep Learning-Based 3D Point Cloud Completion Processing and Analysis. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 22862–22883. [CrossRef]
18. Wang, X.; Ang, M.H.; Hee Lee, G. Voxel-based Network for Shape Completion by Leveraging Edge Generation. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 13169–13178.
19. Junshu, T.; Zhijun, G.; Ran, Y.; Yuan, X.; Lizhuang, M. LAKe-Net: Topology-Aware Point Cloud Completion by Localizing Aligned Keypoints. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 1726–1735.
20. Yingjie, C.; Kwan-Yee, L.; Chao, Z.; Qiang, W.; Xiaogang, W.; Hongsheng, L. Learning a Structured Latent Space for Unsupervised Point Cloud Completion. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 5543–5553.
21. Huang, Z.; Yu, Y.; Xu, J.; Ni, F.; Le, X. PF-Net: Point Fractal Network for 3D Point Cloud Completion. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 7659–7667.
22. Wen, X.; Xiang, P.; Han, Z.; Cao, Y.P.; Wan, P.; Zheng, W.; Liu, Y.S. PMP-Net: Point Cloud Completion by Learning Multi-step Point Moving Paths. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 7439–7448.
23. He, K.; Chen, X.; Xie, S.; Li, Y.; Dollar, P.; Girshick, R. Masked Autoencoders Are Scalable Vision Learners. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2022), New Orleans, LA, USA, 18–24 June 2022; pp. 15979–15988.
24. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
25. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. ShapeNet: An Information-Rich 3D Model Repository. *arXiv* **2015**, arXiv:1512.03012.
26. Lim, I.; Ibing, M.; Kobbelt, L. A Convolutional Decoder for Point Clouds using Adaptive Instance Normalization. *Comput. Graph. Forum* **2019**, *38*, 99–108. [CrossRef]
27. Wang, X.; Ang, M.H.; Lee, G.H. Cascaded Refinement Network for Point Cloud Completion. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 787–796.
28. Zhang, W.; Yan, Q.; Xiao, C. Detail Preserved Point Cloud Completion via Separated Feature Aggregation. In *Computer Vision—ECCV 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., Eds.; Springer: Cham, Switzerland, 2020; pp. 512–528.
29. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
30. Esposito, C.; Landrum, G.A.; Schneider, N.; Stiefl, N.; Riniker, S. GHOST: Adjusting the Decision Threshold to Handle Imbalanced Data in Machine Learning. *J. Chem. Inf. Model.* **2021**, *61*, 2623–2640. [CrossRef] [PubMed]