Point cloud super-resolution based on geometric constraints

Xiaqiang Li¹² | Jitao Liu¹ | Songmin Dai¹

¹School of Computer Engineering and Science, Shanghai University, China
²Shanghai Institute for Advanced Communication and Data Science, Shanghai University, China

Correspondence
Xiaqiang Li
School of Computer Engineering and Science, Shanghai University, China.
Email: xqli@shu.edu.cn

Abstract
Among all digital representations we have for real physical objects, three-dimensional (3D) is arguably the most expressive encoding. But due to the limitations of 3D scanning equipment, point cloud often becomes sparse or partially missing. A point cloud super-resolution (PCSR) method based on geometric constraints is proposed to solve the sparse problem of point clouds: it allows dense point clouds to be generated by sparse point clouds. The method is based on the conditional generative adversarial network including redesigned generator and discriminator for point cloud data specialty. Moreover, the method can maintain the shape of the dense point cloud by adding geometric constraints. The contributions of our work are as follows: (1) a PCSR method based on geometric constraints is proposed; (2) add a module for obtaining point cloud neighbourhood information in the generator, called K-nn operation module; and (3) feature aggregation is performed using the weighted pooling to process the neighbourhood information obtained by the K-nn operation module. Extensive experimental results demonstrate the effectiveness of the proposed method.

1 INTRODUCTION

Among all digital representations we have for real physical objects, three-dimensional (3D) is arguably the most expressive encoding. 3D representations allow storage and manipulation of high-level information (e.g. semantics, affordances, function) as well as low-level features (e.g. appearance, materials) about the object. However, unlike the 2D image, there is no unified output representation for 3D geometry. The choice of representation is critical for learning a good generative model of 3D shapes. Voxel representations are a straightforward generalization of pixels to the 3D case. But its shortcomings are also obvious. Voxel leads to data sparsity and computation cost of 3D convolution. Some methods use meshes to represent the discrete high-dimensional space, so that the objects can be represented naturally. However, these methods require a template measure of the same topology and do not allow arbitrary changes. Point clouds are a set of points; the representation of point clouds is much simpler than that of mesh. Although there is no connection between points in the point cloud, point cloud is most close to raw sensor data.

3D point cloud understanding is a long-standing problem. Typical tasks include 3D object classification [1], 3D object detection [2-5], 3D semantic segmentation [6-8] and 3D object reconstruction [9, 10]. Recently, PointNet [6] architecture directly operates on point cloud instead of the 3D voxel grid or mesh. It not only accelerates computation but also notably improves the segmentation performance. However, PointNet does not capture local structures induced by the metric space points live in, limiting its ability to recognize fine-grained patterns and generalizability to complex scenes. To address this problem, PointNet++ [7] leverages neighbourhoods at multiple scales to achieve both robustness and detail capture. DGCNN [11] proposes a new neural network module dubbed EdgeConv suitable for CNN-based high-level researches. It cannot only incorporate local neighbourhood information but also stack or recurrently apply to learn global shape properties.

The sparse point cloud can describe the outline of the object, but the lack of detail will affect its classification and recognition. In the field of a two-dimensional image, single image super resolution (SISR) is a classic computer vision problem, which aims to recover a high-resolution (HR) image from a low-resolution (LR) image. The popular method is to use the deep convolution neural network and generative adversarial networks to restore the higher quality image. Since SISR can restore high-frequency information, it is widely used in applications such as medical imaging [12], satellite imaging [13], security and surveillance [14], where high-frequency details are
greatly desired. Motivated by these methods, now we can use deep neural networks such as PU-Net [15], EC-Net [16] and PU-GAN [17] to sample the point cloud. However, these networks cannot produce reliable results for extremely sparse and uneven low-quality inputs. To get excellent results, these works also need to provide a lot of label information for the dataset.

As we have known, the point cloud is three-dimensional but two-dimensional. Therefore, it is necessary to design a network different from the general neural network to adapt to the particularity of point cloud format. To address this problem, the Point Cloud Super-Resolution (PCSR) method is proposed in this work. The method uses a conditional generative adversarial network, which is divided into generator and discriminator. Low-resolution point cloud is inputted into the generator to generate high-resolution point cloud. The point cloud pairs are fed into the discriminator to determine whether the high-resolution point cloud is real or not. Our method can learn local and global features at the same time. Global features can guide the generation of point clouds, and local features can improve the details and textures of the results. Some techniques [7, 18] for learning local features treat points independently at a local scale to maintain permutation invariance. This independence, however, neglects the geometric relationships among points, presenting a fundamental limitation that leads to local features missing. Instead of generating points features directly from point cloud itself; our method generates edge features that describe the relationships between a point and its neighbours. The method cannot only capture local geometric structures but also maintain permutation invariance. Moreover, the robustness of the network is improved by the weighted pooling aggregation edge features. In PCSR method, low-resolution point clouds are input to the generator, and high-resolution point clouds are obtained by fusing local and global features. The discriminator determines whether the generated high-resolution point cloud is correct and makes the generated result of the generator better. Extensive experimental results demonstrate the effectiveness of the proposed algorithm.

The rest of this study is organized as follows: In Section 2, we review some traditional approaches for point cloud and image super-resolution. Section 3 gives a detailed explanation of PCSR and the experiment results are shown in Section 4. Finally, this work is concluded in Section 5.

2 | RELATED WORKS

2.1 | Point cloud

Various 3D representations [1, 6, 7] have been explored recently based on deep learning on 3D data. Among them, point cloud representation is becoming increasingly popular due to its memory efficiency and intuitiveness.

A point cloud represents a geometric shape typically its surface as a set of 3D locations in a Euclidean coordinate frame. In 3D space, these locations are defined by their $x$, $y$, $z$ coordinates. Thus, the point cloud representation of an object or scene is an $N \times 3$ matrix, where $N$ is the number of points, referred to as the point cloud.

Some latest works [6, 7, 11, 19] directly take raw point clouds as input without converting them to other formats. PointNet [6] pioneered point clouds as a representation of deep learning tasks. It is a unified architecture that learns both global and local point features, providing a simple, efficient and effective approach for a number of 3D recognition tasks, like classification, part segmentation, semantic segmentation. PointNet proposed to use shared multi-layer perceptrons and max-pooling layers to obtain the feature of point cloud. They achieved permutation invariance by applying a fully connected neural network to each point independently followed by a max-pooling operation. This approach is to approximate a general function defined on a point set by applying a symmetric function on disorder point cloud in the set:

$$f(x_1, \ldots, x_n) \approx g(b(x_1), \ldots, b(x_n))$$ (1)

where $b$ is a multi-layer perceptron network and $g$ is a composition of a single variable function and a max-pooling function. Each point contains 3D coordinates $x_i = (x_i, y_i, z_i)$.

Because the max-pooling layers are applied across all the points in point cloud, it is difficult to capture local features, limiting its ability to recognize fine-grained patterns and generalisability of complex scenes. PointNet++ [7] improved the network in PointNet by adding a hierarchical structure. The hierarchical structure is similar to CNNs used in image CNNs, which extracts features starting from small local regions and gradually extending to larger regions. By applying this structure, PointNet++ is able to learn local features with increasing contextual scales. PointCNN [19] is a generalization of CNN into leveraging spatially local correlation from data represented in point cloud. The core of PointCNN is the X-Conv operator which weights and permutes input points and features before they are processed by a typical convolution. However, PointCNN is unable to achieve permutation-invariance which is necessary for point clouds.

Dynamic graph CNN [11] proposed a method that can dynamically update the graph. This approach is inspired by PointNet and convolution operations. Instead of working on individual points like PointNet, however, graph neural networks exploit local geometric structures by constructing a local neighbourhood graph and applying convolution-like operations on the edges connecting neighbouring pairs of points. Differently from traditional graph CNNs, dynamic graph CNN is dynamically updated after each layer of the network. The proximity in feature space differs from proximity in the input, and the dynamic update leads to nonlocal diffusion of information throughout the point cloud. Wu et al. [20] extend the dynamic filter to a new convolution operation, named PointConv. PointConv can be applied on point clouds to build deep convolutional networks. This approach treats convolution kernels as nonlinear functions of the local coordinates of 3D points comprising weight and density functions. With respect to a given point, the weight functions are learnt with multi-layer perceptron networks and density functions are learnt through kernel density estimation.
2.2 | Super-resolution

Many image super-resolution methods have been developed in the computer vision community. The pioneer work is termed as the Super-Resolution Convolutional Neural Network (SRCNN) proposed by Dong et al. [21]. SRCNN first upscale low-resolution images to the desired size using bicubic interpolation, which is the only pre-processing SRCNN perform. SRCNN’s network structure is very simple, using only three convolution layers. However, the high computational cost still hinders it from practical usage that demands real-time performance. FSRCNN [22] aims at accelerating the current SRCNN and proposes a compact hourglass-shape CNN structure for faster and better SR. FSRCNN redesign the SRCNN structure mainly in three aspects. Subsequently, a deep network with 20 layers was proposed in VDSR [23] to improve the reconstruction accuracy of CNN. VDSR uses a very deep convolutional network inspired by VGG-net used for ImageNet classification [24]. Training a very deep network is hard due to a slow convergence rate. The residuals between the HR images and the interpolated LR images were used in [23] to speed up the converging speed in training and also to improve the reconstruction performance. Tai et al. [25] propose Deep Recursive Residual Network (DRRN) for SISR. In DRRN, an enhanced residual unit structure is recursively learnt in a recursive block and it stacks several recursive blocks to learn the residual image between the HR and LR images. The residual image is then added to the input LR image from a global identity branch to estimate the HR image. In SRDenseNet [26], the feature maps of each layer are propagated into all subsequent layers, providing an effective way to combine the low-level features and high-level features to boost the reconstruction performance. Ledig et al. [27] propose a super-resolution generative adversarial network (SRGAN) for which it employs a deep residual network (ResNet) with skip-connection and diverges from MSE as the sole optimization target. Different from previous works, SRGAN proposes a novel perceptual loss using high-level feature maps of the VGG network combined with a discriminator that encourages solutions perceptually hard to distinguish from the HR reference images.

Different from image super-resolution, Yu et al. [15] introduced a deep neural network PU-Net to upsampling point sets based on the PointNet++ architecture. The core idea is to learn the multi-level features of each point and implicitly expand the point set through the multi-branch convolution unit in the feature space. The expanded feature is divided into multiple features and these features are reconstructed into an up-sampling point set. PU-Net is applied at a patch level and has a joint loss function that encourages the upsampling points to remain uniformly distributed on the underlying surface. However, PU-Net cannot detect the edge on the regular object point cloud, which leads to the uneven edges and corners of the object and presents the defect of irregular undulation. Later, they introduced EC-Net [16] for edge-aware point cloud upsampling by formulating an edge-aware joint loss function to minimize the point-to-edge distances. EC-Net design network to process points grouped in local patches, and train it to learn and help consolidate points, deliberately for edges. Through this method, EC-Net can attend to the detected sharp edges and enable more accurate 3D reconstructions in the process of sampling on the point cloud. However, these thresholds and edges are in fact manually marked on the point cloud in advance, resulting in the data that needs to be manually labelled. PU-GAN [17] proposes a generative adversarial network to sample on point cloud. To realize a working GAN network, PU-GAN constructs an up-down-up expansion unit in the generator for upsampling point features with error feedback and self-correction and formulate a self-attention unit to enhance the feature integration. Further, it designs a compound loss with adversarial, uniform and reconstruction terms, to encourage the discriminator to learn more latent patterns and enhance the output point distribution uniformity. But it still uses the architecture of PointNet++, the feature extraction and fusion in feature space is not enough.

3 | METHOD

We propose a PCSR method based on geometric constraints using a conditional generative adversarial network. The method is divided into generator and discriminator. Consider a low-resolution (LR) point cloud with 1024 points, the ground truth high-resolution (HR) point cloud contains 4096 points. The ground truth is downsampled to 1024 points, then super-resolution to 4096 points. This can better evaluate the super-resolution result. We still call the super-resolution point cloud a “low-resolution” point cloud, although it has the same number of points as the high-resolution point cloud.

LR point cloud is inputted into generator $G$ to generate HR point clouds. The point cloud pairs are fed into the discriminator $D$ to determine whether the HR point cloud is real or not. An overview of PCSR generative adversarial network is shown in Figure 1.

3.1 | The architecture of generator

The generator architecture is visualized in Figure 2. The generator has four key blocks: one joint alignment networks block named spatial transform network that align input points, $K$-nn operation block get point cloud $K$ nearest neighbours, weighted pooling block aggregate the features of $K$ neighbours by weighted averaging and a local and global information combination block .

We discuss the reason behind these design choices in separate paragraphs below.

**Spatial Transform Network Block**: The semantic labeling of a point cloud needs to be invariant if the point cloud undergoes certain geometric transformations, such as rigid transformations. It is therefore expected that the learnt representation of point set is invariant to these transformations. Inspired by Jaderberg et al. [28], we predict an affine transformation matrix by spatial transform network and directly
apply this transformation to the coordinates of input points. The spatial transform network resembles the traditional network and is composed by basic modules of point independent feature extraction, max pooling and fully connected layers.

**K-nn Operation Block:** Local information is important for feature learning. Some approaches only working on individual points like PointNet, in contrast, we exploit local geometric structures by getting the nearest neighbour of the centre point and apply convolution-like operations on the

---

**Figure 1** The overview of point cloud super-resolution based on the conditional generative adversarial network. Low-resolution point cloud is inputted into the generator to generate a high-resolution point cloud. The discriminator determines whether the high-resolution point cloud is real and conforms to the input.

**Figure 2** The architecture of generator. The model takes as input $n$ points, calculates an edge feature set of size $K$ for each point at a K-nn operation and aggregates features by weighted pooling. Finally, the high-resolution point cloud is acquired by concatenating the global descriptor and local descriptors. **Spatial transform block:** The spatial transform block is designed to align an input point set to a canonical space by applying an estimated $3 \times 3$ matrix. **K-nn operation block:** The K-nn operation block exploits local geometric structures by getting the nearest neighbour of the centre point and applying convolution-like operations on the edges connecting neighbouring pairs of points. **Weighted pooling block:** The Weighted pooling block aggregates features through the importance of neighbours.
edges connecting neighbouring pairs of points. Consider an F-dimensional point cloud with \( n \) points, \( X = \{ x_1, \ldots, x_i, \ldots, x_n \} \subseteq \mathbb{R}^F \). In the simplest setting, \( F = 3 \), it is also possible to include additional coordinates representing colour, surface normal and so on. We use the \( k \)-nearest neighbour algorithm to search points \( x_{j1}, x_{j2}, \ldots, x_{jk} \) and make these points closest to \( x_i \). Then, we calculate \( x_j - x_i \), this operation encodes local information. However, if the PCSR method only encodes local information, it will treat the shape as a collection of small patches and lose the global shape structure. So, we finally combine both the global shape structure (captured by the coordinates of the patch centres \( x_j \)) and local neighbourhood information (captured by \( x_j - x_i \)).

**Weighted Pooling Block:** DGCNN [11] applies a channel-wise symmetric aggregation operation (max pooling) on the edge features associated with all the edges emanating from each vertex. Different from [11], we think that the distance between neighbours has different effects on the central point. This is because using max pooling as a symmetric aggregation operation only retrieves information from the farthest neighbours. So we design a weighted pooling operation to solve this problem. After search neighbour points \( x_{j1}, x_{j2}, \ldots, x_{jk} \) of the centre point \( x_i \), the distance \( d_{j1}, d_{j2}, \ldots, d_{jk} \) between neighbour points and the centre point is calculated. The weight \( \alpha \) of each point by distance is defined as follows:

\[
\alpha_i = \frac{e^{-[(x_i-x_{j1})^2+(x_i-x_{j2})^2+(x_i-x_{jk})^2]}}{\sum_{n=1}^{k} e^{-[(x_i-x_{n1})^2+(x_i-x_{n2})^2+(x_i-x_{nk})^2]}}
\]  

(2)

The weighted pooling value \( \beta \) is defined as follows:

\[
\beta = \sum_{i=1}^{k} \alpha_i x_{ji} = \frac{\sum_{i=1}^{k} e^{-[(x_i-x_{j1})^2+(x_i-x_{j2})^2+(x_i-x_{jk})^2]} x_{ji}}{\sum_{i=1}^{k} e^{-[(x_i-x_{n1})^2+(x_i-x_{n2})^2+(x_i-x_{nk})^2]}}
\]  

(3)

**Local and Global Information Aggregation:** To make a model invariant to input permutation, we use a simple symmetric function to aggregate the information from each point. Here, a symmetric function takes all \( n \) vectors as input and output a new vector that is invariant to the input order. The output of the symmetric function is a global signature of the input set. PCSR requires a combination of local and global knowledge. Our solution can be seen in Figure 2. After computing the global point cloud feature vector, the generator feeds it back to per point features by concatenating the global feature with each of the point features. Then, the generator extracts a new feature of each point based on the combined point features—this time each point feature is aware of both the local and global information.

In Figure 2, after the spatial transform network block and K-NN operation block, there are three Conv blocks. The first Conv block uses two multi-layer perceptron layers (64,64). The second Conv block uses two multi-layer perceptron layers (128,128) while the third Conv block uses two multi-layer perceptron layer (256). After obtaining local features, we use one Conv block with a multi-layer perceptron layer (1024) to extract the feature. Finally, we use a Conv block with five multi-layer perceptron layers (512,256,128,128,3) to obtain high-resolution point cloud.

**3.2 The architecture of discriminator**

Inspired by [29], the proposed method guides the data generation process by adding condition constraints to the discriminator. The discriminator architecture is visualized in Figure 3. The low-resolution point cloud and the high-resolution point cloud are inputted into the discriminator at the same time. This method allows the discriminator to discern whether the generated output is true and whether it matches the generator’s input. As a result, the output can maintain the same distribution as the original input.

Same as the generator, the discriminator uses the spatial transform network to remain invariant if the point cloud undergoes certain geometric transformations. This idea can be further extended to the alignment of the feature space.
Another alignment network named feature transformations network is inserted on the point feature map and predict a feature transformation matrix to align features from different input point clouds.

After getting the global feature by symmetric function, the multi-layer perceptron classifier can be trained on the global features for classification.

The discriminator consists of three Conv blocks. The first Conv block uses two multi-layer perceptron layers (64, 64). The second Conv block uses three multi-layer perceptron layers (128, 512, 1024) while the third Conv block uses a multi-layer perceptron layer (512, 256, 1).

### 3.3 Objective function

The objective of Point Cloud Colourization Network can be expressed as:

$$L_{GAN}(D, G) = E_{xy}[\log D(x, y)] + E_{xy}[\log(1 - D(x, G(x, z)))]$$

(4)

where $G$ tries to minimize this objective against an adversarial $D$ which tries to maximize it.

A critical challenge is to design a good loss function for comparing the predicted point cloud and the ground-truth. To plug in a neural network, a suitable distance must satisfy at least three conditions: (1) differentiable with respect to point locations; (2) efficient to compute, as data will be forwarded and back-propagated for many times; (3) robust against the small number of outlier points in the sets. So we use Chamfer distance (CD) as a loss function. For each point, the algorithm of CD finds the nearest neighbor in the other set and sums the squared distances up.

$$L_{CD}(G) = \sum_{x \in G(x, y)} \min_{y \in y} ||\hat{x} - \hat{y}||^2 + \sum_{y \in y} \min_{x \in G(x, y)} ||\hat{x} - \hat{y}||^2$$

(5)

Therefore, our final objective is

$$L_{PCSR}(D, G) = L_{GAN}(D, G) + \lambda L_{CD}(G).$$

(6)

Our loss function contains $L_{GAN}(D, G)$ and $L_{CD}(G)$. Parameter $\lambda$ is used to adjust the proportion of $L_{CD}(G)$.

### 3.4 Implementation details

Generative adversarial networks (GANs) [30] have been enjoying considerable success as a framework of generative models in recent years, and they have been applied to numerous types of tasks and datasets [31, 32]. A persisting challenge in the training of GANs is the performance control of the discriminator. In high dimensional spaces, the density estimation by the discriminator is often inaccurate and unstable during the training, and the generator fails to learn the multi-modal structure of the target distribution. To solve this problem, we draw on the experience of SNGAN [33] to stabilize the training of discriminator networks. The Lipschitz constant is the only hyper-parameter to be tuned in SNGAN, and the algorithm does not require intensive tuning of the only hyper-parameter for satisfactory performance. Compared with other methods, its implementation is simple and the additional computational cost is small.

Batch normalization (BN) is a milestone technique in the development of deep learning. BN normalizes the features by the mean and variance computed within a (mini-)batch. This has been shown by many practices to ease optimization and enable very deep networks to converge. However, normalizing along the batch dimension introduces problems—BN's error increases rapidly when the batch size becomes smaller, caused by inaccurate batch statistics estimation. Due to the limitation of memory, only a small batch size can be used for PCSR. To avoid this problem, we use group normalization (GN) [34] as a simple alternative to BN. [34] notices that many classical features like SIFT [35] and HOG [36] are group-wise features and involve group-wise normalization. For example, an HOG vector is the outcome of several spatial cells where each cell is represented by a normalized orientation histogram. Analogously, GN divides the channels into groups and computes within each group the mean and variance for normalization. GN's computation is independent of batch sizes, and its accuracy is stable in a wide range of batch sizes.

### 4 EXPERIMENTS

#### 4.1 Datasets and preprocessing

We use the ShapeNetCore subset of the ShapeNet [37] as our dataset. ShapeNet is a richly annotated, large scale dataset of 3D shapes. ShapeNetCore is a densely annotated subset of ShapeNet covering 55 common object categories with about 51,300 unique 3D models. To satisfy the network input format, we convert the OBJ file in the dataset into the point cloud format and then perform uniform sampling to obtain low-resolution point cloud and high-resolution point cloud, which is shown in Figure 4.

#### 4.2 Experimental results of point cloud super-resolution

The network is trained to use a subset of ShapenetCore in ShapeNet. Some generated samples are visualized in Figure 5. The point cloud in the left column is the low-resolution point cloud that is inputted to the generator. The point cloud in the middle column is the super-resolution point cloud that is outputted from the generator. The point cloud in the right column is ground truth.
4.3 Comparison with other methods

To verify the rationality of our structure, we design some baseline methods for comparison. The experiment is conducted based on the proposed method PCSR, original PointNet [6], PointNet++ [7], PointCNN [19], DGCNN [11] and SAWNet [38] architecture. And our model is compared with PU-Net [15] and PU-GAN [17]. For PU-Net and PU-GAN, we used their public code and retrained their networks using our training data, the results are listed in Figure 6.

To quantitatively evaluate the quality of the output point sets, we formulate two metrics to measure the deviation between the output points $S_1$ and the ground truth meshes $S_2$, as well as the distribution uniformity of the output points.

For Chamfer distance (CD) in Equation (7), the algorithm finds the nearest neighbour in the other set and sums the squared distances up.

$$D_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} ||x - y||_2^2 + \sum_{y \in S_2} \min_{x \in S_1} ||x - y||_2^2$$  \hspace{1cm} (7)

The second metric is Earth Mover’s distance (EMD) [39]:

$$D_{EMD}(S_1, S_2) = \min_{\phi: S_1 \rightarrow S_2} \sum_{x \in S_1} ||x - \phi(x)||_2$$  \hspace{1cm} (8)

where $\phi: S_1 \rightarrow S_2$ is a bijection. The EMD distance solves an optimization problem, namely, the assignment problem. For all but a zero measure subset of point set pairs, the optimal bijection $\phi$ is unique and invariant under infinitesimal movement of the points.

Tables 1 and 2 list two kinds of quantitative comparison results on ShapeNet dataset, respectively. It can be seen that the original PointNet cannot generate the point cloud. This is because PointNet has no local information to guide the generation of point cloud. PointNet++ and PointCNN solve the problem of PointNet by getting the information of neighbour points, but they do not have a good feature fusion of point cloud information. The super-resolution point cloud generated by DGGNN does not have very good constraints on the distribution of point clouds. This is because DGCNN chose the
maximum aggregation mode for neighbours and had not obtained local information very accurately. SAWNet combines PointNet and DGCNN to improve the complexity of the network, but it will lead to over-fitting and increase the training time. Compared with baseline methods, the results of PU-Net and PU-GAN are greatly improved. But PU-Net and PU-GAN are based on the first extraction of patches, the network cannot fully extract features and reconstruct the data with low mesh quality, such as ShapeNet. However, the PCSR model can generate the super-resolution point cloud very well;

| Method   | Mean | Car   | Guitar | Knife | Skateboard | Bag   | Chair | Computer | Lamp   | Plane | Table |
|----------|------|-------|--------|-------|------------|-------|-------|----------|--------|-------|-------|
| PointNet | 3.86 | 1.03  | 4.39   | 3.64  | 5.22       | 1.03  | 4.09  | 3.51     | 4.16   | 4.84  | 3.15  |
| PointNet++| 0.01018 | 0.00565 | 0.00302 | 0.0544 | 0.04141   | 0.0151 | 0.00727 | 0.06243 | 0.0184 | 0.00323 | 0.0059 |
| PointCNN | 0.01163 | 0.00505 | 0.00187 | 0.0283 | 0.03634   | 0.0209 | 0.00632 | 0.08577 | 0.01695 | 0.00211 | 0.00537 |
| DGCNN    | 0.00908 | 0.00723 | 0.00270 | 0.01560 | 0.00742   | 0.02050 | 0.00968 | 0.01710 | 0.01730 | 0.00440 | 0.00848 |
| SAWNet   | 0.00712 | 0.00124 | 0.00222 | 0.01710 | 0.00205   | 0.03750 | 0.00804 | 0.08290 | 0.04400 | 0.00136 | 0.00620 |
| PU-Net   | 0.00088 | 0.00093 | 0.00054 | 0.00034 | 0.00086   | 0.00128 | 0.00072 | 0.00136 | 0.00526 | 0.00061 | 0.00090 |
| PU-GAN   | 0.00076 | 0.00089 | 0.00072 | 0.00019 | 0.00028   | 0.00184 | 0.00081 | 0.00080 | 0.00704 | 0.00050 | 0.00069 |
| Our      | 0.00042 | 0.00055 | 0.00015 | 0.00017 | 0.00044   | 0.00172 | 0.00040 | 0.00080 | 0.00154 | 0.00011 | 0.00030 |

**Table 1** Chamfer distance result on ShapeNet

**Figure 6** Comparison of our model with other methods. Experimental results show that our architecture works better.
this is because PCSR model gets the local information by K-nn operation and aggregate the local information by weighted pooling. This method can accurately learn the distribution of point clouds.

### 4.4 | Ablation experiments

#### 4.4.1 | Comparison with different aggregation mode

In this section, we compare the aggregation modes which are max pooling, mean pooling and weighted pooling on the three categories. The results are shown in Figure 7. From the experiment, it can be concluded that the results generated using max-pooling and mean pooling cannot well express the shape of the object. This is because the maximum pooling gets the farthest point in the neighbourhood, and the average pooling gets the centre of gravity of the neighbourhood. These two methods cannot get the neighbour information around the centre point very well, so the experimental results are not good enough. On the contrary, the results of the weighted pooling proposed by this paper can better maintain the shape of the object. This verifies our previous conjecture in Section 3.1.

#### 4.4.2 | Comparison of different neighbourhood numbers

In order to find out whether the number of neighbourhoods will affect the generated point cloud, we set K to 10, 20, 30. We selected three categories in the ShapeNetCore dataset to complete the experiment.

Experiment results are shown in Figure 8, it can be seen that the best result is generated when $K = 20$. This is because when $K = 10$, the neighbour's information is not enough to express the local information completely. Additional, too much useless neighbour information will be added when $K = 30$ so that there are many interference information in the features. And the greater the K, the longer the training time will be. So $K = 20$ is used as the number of neighbours.

### 5 | CONCLUSION

We propose a conditional adversarial network with point cloud feature extraction for PCSR. The generator and discriminator of the network are redesigned to be suitable for the point cloud.
This architecture can capture the global and local features to maintain the shape of the super-resolution point cloud. In order to restore the details of the super-resolution point cloud, geometric constraints are added. The geometric constraints are achieved using K-nn operation to obtain the neighbour features of each point and using the new pooling method—weighted pooling to aggregate features. Moreover, the structure improvement of the conditional adversarial network also makes the training process more stable. Experimental results show that the proposed network is reasonable for PCSR.

**REFERENCES**

1. Wu, Z., et al.: 3dShapeNets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1912–1920 (2015)
2. Lahoud, J., Ghanem, B.: 2d-driven 3d object detection in RGB-D images. In: Proceedings of the IEEE conference on computer vision, pp. 4622–4630 (2017)
3. Qi, C.R., et al.: Frustum PointNets for 3d object detection from RGB-D data. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 918–927 (2018)
4. Song, S., Xiao, J.: Deep sliding shapes for amodal 3d object detection in RGB-D images. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 808–816 (2016)
5. Shi, S., Wang, X., Li, H.: PointRCNN: 3d object proposal generation and detection from point cloud. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–779 (2019)
6. Qi, C.R., et al.: PointNet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 652–660 (2017)
7. Qi, C.R., et al.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems, pp. 5099–5108 (2017)
8. Riegler, G., Osman Ulusoy, A., Geiger, A.: OctNet: Learning deep 3d representations at high resolutions. In: In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3577–3586 (2017)
9. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 605–613 (2017)
10. Lin, C.H., Kong, C., Lucey, S.: Learning efficient point cloud generation for dense 3d object reconstruction. In: Thirty-Second AAAI conference on artificial intelligence (2018)
11. Wang, Y., et al.: Dynamic graph cnn for learning on point clouds[1]. Acm Transactions On Graphics (tog). Dynamic graph cnn for learning on point clouds 38(5), 1–12 (2019) arXiv preprint arXiv:180708729
12. Shi, W., et al.: Cardiac image super-resolution with global correspondence using multi-atlas patchmatch. In: In International conference on medical image computing and computer-assisted intervention, pp. 9–16. Springer, Berlin (2013)
13. Thornton, M.W., Atkinson, P.M., Holland, D.: Sub-pixel mapping of rural land cover objects from fine spatial resolution satellite sensor imagery using super-resolution pixel-swapping. Intl. J. Rem. Sems 27(3), 473–491 (2006)
14. Zou, W.W., Yuen, P.C.: Very low resolution face recognition problem. IEEE Trans. Image Process 21(1), 327–340 (2011)
15. Yu, L., et al.: Pu-net: point cloud upsampling network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2790–2799 (2018)
16. Yu, L., et al.: Ec-net: an edge-aware point set consolidation network. In: Proceedings of the European conference on computer vision (ECCV), pp. 386–402 (2018)
17. Li, R., et al.: Pu-gen: a point cloud upsampling adversarial network. In: Proceedings of the IEEE international conference on computer vision, pp. 7203–7212 (2019)
18. Shen, Y., Feng, C., Yang, Y., et al.: Neighbors do help: Deeply exploiting local structures of point clouds, vol. 1(2), (2017) arXiv preprint arXiv:171206760. https://arxiv.org/abs/1712.06760
19. Li, Y., et al.: PointCNN: Convolution on x-transformed points. In: Advances in neural information processing systems, pp. 820–830 (2018)
20. Wu, W., Qi, Z., Fuxin, L.: Pointconv: Deep convolutional networks on 3d point clouds. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 9621–9630 (2019)
21. Dong, C., et al.: Learning a deep convolutional network for image super-resolution. In: European conference on computer vision, pp. 184–199. Springer, Berlin (2014)
22. Dong, C., Loy, C.C., Tang, X.: Accelerating the super-resolution convolutional neural network. In: European conference on computer vision, pp. 391–407. Springer, Berlin (2016)
23. Kim, J., Kwon Lee, J., Mu Lee, K.: Accurate image super-resolution using very deep convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1646–1654 (2016)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: In Proceedings of international conference of learning representation (2015)
25. Tai, Y., Yang, J., Liu, X.: Image super-resolution via deep recursive residual network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3147–3155 (2017)
26. Tong, T., et al.: Image super-resolution using dense skip connections. In: Proceedings of the IEEE international conference on computer vision, pp. 4799–4807 (2017)
27. Ledig, C., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4681–4690 (2017)
28. Jaderberg, M., et al.: Spatial transformer networks. In: Advances in neural information processing systems, pp. 2017–2025 (2015)
29. Mirza, M., Osindero, S.: Conditional generative adversarial nets (2014) arXiv preprint arXiv:1411.1784
30. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in neural information processing systems, pp. 2672–2680 (2014)
31. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. (2015). https://arxiv.org/abs/1511.06434
32. Salimans, T., et al.: Improved techniques for training gans. In: Advances in neural information processing systems, pp. 2234–2242 (2016)
33. Miyato, T., Kataoka, T.,Koyama, M., et al.: Spectral normalization for generative adversarial networks. (2018). https://arxiv.org/abs/1802.05957
34. Wu, Y., He, K.: Group normalization. In: Proceedings of the European conference on computer vision (ECCV), pp. 3–19 (2018)
35. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Intl. J. Comput. Vis. 60(2), 91–110 (2004)
36. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), vol. 1, pp. 886–893. San Diego (2005). https://doi.org/10.1109/CVPR.2005.177
37. Chang, A.X., Funkhouser, T., Guibas, L., et al.: ShapeNet: An information-rich 3d model repository. (2015). https://arxiv.org/abs/1512.03012
38. Kaul, C., Pears, N., Manandhar, S.: SAWNet: A spatially aware deep neural network for 3d point cloud processing. (2019). https://arxiv.org/abs/1905.07650
39. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover’s distance as a metric for image retrieval. Intl. J. Comput. Vis. 40(2), 99–121 (2000)

**How to cite this article:** Li X, Liu J, Dai S. Point cloud super-resolution based on geometric constraints. *IET Comput. Vis.* 2021;15:312–321. https://doi.org/10.1049/cvi.12.045