applied sciences

MDPI

Article

RE-PU: A Self-Supervised Arbitrary-Scale Point Cloud Upsampling Method Based on Reconstruction

Yazhen Han [1], Mengxiao Yin [1,2,*], Feng Yang [1,2] and Feng Zhan [1,2]

[1] School of Computer and Electronics Information, Guangxi University, Nanning 530004, China; 2013301011@st.gxu.edu.cn (H.Y.); yf@gxu.edu.cn (F.Y.); zf@gxu.edu.cn (F.Z.)
[2] Guangxi Key Laboratory of Multimedia Communications and Network Technology, Nanning 530004, China
* Correspondence: ymx@gxu.edu.cn

Abstract: The point clouds obtained directly from three-dimensional scanning devices are often sparse and noisy. Therefore, point cloud upsampling plays an increasingly crucial role in fields such as point cloud reconstruction and rendering. However, point cloud upsampling methods are primarily supervised and fixed-rate, which restricts their applicability in various scenarios. In this paper, we propose a novel point cloud upsampling method, named RE-PU, which is based on the point cloud reconstruction and achieves self-supervised upsampling at arbitrary rates. The proposed method consists of two main stages: the first stage is to train a network to reconstruct the original point cloud from a prior distribution, and the second stage is to upsample the point cloud data by increasing the number of sampled points on the prior distribution with the trained model. The experimental results demonstrate that the proposed method can achieve comparable outcomes to supervised methods in terms of both visual quality and quantitative metrics.

Keywords: point cloud; point cloud upsampling; point cloud reconstruction

Citation: Han, Y.; Yin, M.; Yang, F.; Zhan, F. RE-PU: A Self-Supervised Arbitrary-Scale Point Cloud Upsampling Method Based on Reconstruction. *Appl. Sci.* **2024**, *14*, 6814. https://doi.org/10.3390/app14156814

Academic Editor: Andrea Prati

Received: 9 July 2024
Revised: 28 July 2024
Accepted: 30 July 2024
Published: 5 August 2024

1. Introduction

A point cloud is a collection of points in a three-dimensional coordinate system used to represent the surface of objects or scenes. It serves as the standard output of 3D scanning devices such as laser scanners, structured light cameras, and depth cameras widely applied in fields like autonomous driving, robotics, virtual reality, 3D printing, and computer graphics [1]. However, point clouds obtained directly from 3D devices are often sparse, noisy, and uneven, and they may lose information due to occlusion. These challenges pose difficulties for downstream tasks such as rendering and classification. Therefore, point cloud upsampling has garnered increasing attention in recent years, serving as a crucial step to obtain more detailed and accurate scene representations [2].

Point cloud upsampling aims to increase the density of a point cloud, transforming a sparse and uneven low-quality point cloud into a dense and uniform one while preserving the basic geometric shape represented by the original point cloud [3]. It involves generating new points consistent with the original geometry while avoiding distortions like noise and artifacts. Increasing the point cloud can improve the quality of the point cloud and make it more detailed. This is very important for many applications. For example, in the field of rendering, increasing the point cloud can help the model to render more realistic images. In the field of 3D object detection, increasing the point cloud can help the model to detect smaller objects. Point cloud upsampling is a challenging task due to the unordered, sparse, and irregular nature of point clouds [4].

In the early stages, point cloud upsampling was typically achieved through optimization-based methods using various shape priors to refine the point cloud generation, such as normals and local smoothness [5]. However, the effectiveness of these methods was frequently hindered by the intricate structure of 3D models. With the progression of deep learning, PointNet [6] has emerged as a groundbreaking method that directly employs

neural networks for point cloud processing. In recent years, inspired by image super-resolution [7], there has been a growing number of data-driven point cloud upsampling methods proposed, gradually becoming mainstream solutions.

Historically, deep learning based point cloud upsampling methods were primarily supervised and fixed-rate [8,9]. Supervised methods face challenges since they depend on training and testing data with similar distributions. This can be especially challenging in practical applications where achieving identical distributions proves to be difficult. For instance, models trained on synthetic datasets might struggle to perform optimally on real scanned point clouds. On the other hand, fixed-rate upsampling necessitates training separate models for different upsampling rates, leading to considerable inconvenience in practical applications. Additionally, post-processing steps like furthest point sampling were often needed before downstream tasks. Both these characteristics restricted the applicability of conventional methods in various scenarios [10].

In this study, we introduce a novel reconstruction-based point cloud upsampling approach, named RE-PU, which achieves self-supervised upsampling at arbitrary rates. The core idea of this method is to map a prior distribution onto the three-dimensional surface represented by the point cloud. Specifically, we employ an autoencoder or a decoder for self-supervision: the former is for multiple shapes in one model, and the latter is for only one shape. The encoder identifies input point clouds, and the decoder takes the encoder's output along with a prior distribution as input to reconstruct a prior distribution into the input point cloud. Therefore, we can obtain a parametric neural representation of the point cloud surface, which converts a discrete representation into a continuous representation. Subsequently, point cloud upsampling can be achieved by sampling from the prior distribution. By exclusively using test point clouds for training, we avoid the need for constructing paired datasets as training data, thereby mitigating the potential issue of significant differences between training and test data. The ability to sample any number of points from the prior distribution enables achieving point cloud upsampling at arbitrary rates.

More specifically, we introduce a novel neural network for point cloud reconstruction and upsampling. The encoder employs a multi-layer graph convolutional structure [11], dynamically constructing KNN graphs at each layer to increase the network's receptive field, obtain multiscale information, and capture long-distance semantic features at higher layers. It can adaptively acquire local geometric details based on the specific shape of the point cloud. The decoder employs a module based on offset attention [12] for point cloud reconstruction. Offset attention using the Laplacian operator sharpens attention weights compared to typical self-attention mechanisms. This allows for improved preservation of point cloud details and reduces the impact of noise. The prior distribution can be a unit square, a unit sphere, or something else. We conducted extensive experiments with various prior distributions. The results indicate that despite the absence of ground truth (GT), we achieved comparable outcomes to supervised methods. The source code of the proposed method is available at https://github.com/YazhenHan/RE-PU, accessed on 29 July 2024.

The main contributions of this paper are as follows:

- We propose a novel reconstruction-based point cloud upsampling framework.
- We introduce a prior-based point cloud processing network, which can be utilized for both reconstruction and upsampling.
- We demonstrate that the proposed method achieves comparable results to the state-of-the-art methods in terms of both visual quality and quantitative metrics.

2. Related Work

2.1. Deep Learning-Based Point Cloud Upsampling

Point cloud upsampling can be intuitively classified in two ways. Firstly, based to whether dense point clouds are required as labels for sparse point clouds during training, it can be categorized into supervised point cloud upsampling and unsupervised point cloud upsampling. Additionally, based on whether different models need to be trained

for different upsampling factors, it can be classified into arbitrary magnification factor upsampling and fixed magnification factor upsampling. Early methods primarily focused on supervised and fixed magnification factor point cloud upsampling, and we call these methods classical point cloud upsampling methods in this paper. Recently, some unsupervised and arbitrary magnification factor point cloud upsampling methods have emerged, which we call modern point cloud upsampling methods in this paper.

A. Classical point cloud upsampling

Classical point cloud upsampling networks generally consist of three components: a feature extraction module, a feature expansion module, and a point cloud reconstruction module [13]. The feature extraction module encodes the point cloud from three-dimensional Euclidean space into a high-dimensional feature space. The feature expansion module increases the number of features in the point cloud to the required amount. Due to the one-to-one correspondence between point cloud features and point cloud coordinates, the point cloud can be mapped back to the coordinate space from the feature space through the point cloud reconstruction module.

Initially, Yu et al. introduced the first deep learning-based point cloud upsampling method PU-Net [14] based on PointNet+ [15]. This network learns multi-scale features for each point and extends the point set in feature space through a multi-branch convolution. Wang et al. employed dense connections to extract point cloud features and proposed a progressive network named MPU [16], which iteratively refines the input point cloud in multiple steps. It decomposes higher upsampling rates into stepwise doubling and avoids point clustering by concatenating a one-dimensional encoding to capture features of multiple details in the point cloud. Qian et al. introduced PU-GCN [17] based on graph convolution, treating the point cloud as a graph with points as nodes. They utilized a novel inception-based module to extract multiscale information and proposed a NodeShuffle module for feature expansion. Viewing upsampling as a multi-objective task, Li et al. presented Dis-PU [18], a method consisting of two cascaded sub-networks. The former sub-network generates coarse but densely distributed points, while the latter sub-network refines the generated point cloud by adjusting the positions of the points. Long et al. investigated connections between different regions of the point cloud and among individual points, proposing PC2-PU [19]. This method incorporates adjacent point cloud patches as supplementary information into the neural network, requiring additional labels for adjacent point clouds. Combining graph filtering and channel-wise attention, Wang et al. proposed a structure-sensitive upsampling algorithm [20]. Zhao et al. argued that existing upsampling techniques directly learn the mapping from sparse point sets to dense point sets, which is often uncertain and ill-posed. To alleviate the uncertainty and ambiguity in the upsampling mapping, they introduced a universal three-stage vector quantization framework comprising a Codebook Lookup Transformer and knowledge distillation for point cloud upsampling, named CPU [21].

In addition to the typical three-stage approach mentioned above, some innovative methods have incorporated discrete differential geometry into point cloud upsampling, enabling the formulation of upsampling problems through mathematical expressions and providing additional mathematical assurances [22–24]. Moreover, point cloud upsampling can be viewed as a conditional generative task: given a sparse point cloud, generate a corresponding dense point cloud. Therefore, some approaches choose to integrate point cloud upsampling with Generative Adversarial Networks (GANs) [25–27].

In general, supervised point cloud upsampling methods require that the training and testing data conform to similar distributions. For instance, models trained on synthetic datasets may face challenges in achieving optimal performance when applied to point clouds obtained from real-world scans. Moreover, adopting a fixed magnification factor for upsampling entails the necessity of training separate models tailored to different upsampling rates. This introduces notable inconvenience for practical applications. Furthermore, downstream tasks often require additional post-processing steps, such as

furthest point sampling. These traits limit the applicability of conventional methods across diverse scenarios.

B. **Modern point cloud upsampling**

Modern point cloud upsampling methods have emerged in recent years that are unsupervised or arbitrary magnification factor point cloud upsampling methods. These methods do not require paired datasets as training data or can upsample point clouds at arbitrary rates [28,29].

In the context of unsupervised point cloud upsampling, some methods have been proposed to generate point clouds without the need for paired datasets. Zhao et al. introduced a self-supervised point cloud upsampling method SSPU-Net based on differentiable rendering [30]. By leveraging image consistency as a supervisory signal, the network parameters are updated through differentiable rendering. SSAS [31] views point cloud upsampling as the task of finding projection points on the implicit surface. The approach selects seed points, projects them onto the original surface, fits the projection direction and distance using two preceding subtasks, and achieves arbitrary-rate upsampling through furthest point sampling. SPU-Net [32] presents an upsampling framework from coarse to refined, involving downsampling the original point cloud, recovering the original point cloud through feature extraction and expansion, and introducing a self-projection loss to reduce noise in the generated point cloud.

In the context of arbitrary magnification factor point cloud upsampling, some methods have been proposed to upsample point clouds at arbitrary rates using a single model. Inspired by Meta-SR [33] in image super-resolution, Ye et al. proposed Meta-PU [34], a method that dynamically adjusts the upsampling network's weights based on different magnification factors using a sub-network. Feng et al. introduced a novel point cloud representation called Neural Points [35], where each point represents local continuous geometry through a neural field, offering enhanced representational capacity. Mao et al. integrated normalizing flows and point cloud upsampling in PU-Flow [36]. Leveraging the reversible nature of normalizing flows, PU-Flow achieves transformations between 3D Euclidean space and feature space. Mao et al. propose PU-INN [37], a novel reversible residual neural network that allows for an unconstrained architecture design to learn more expressive feature transformations. Grad-PU [38] interpolates low-resolution point clouds based on the given upsampling rate and refines the interpolated point positions through iterative optimization.

In this paper, we propose a novel point cloud upsampling method, named RE-PU, which is based on point cloud reconstruction, achieving self-supervised upsampling at arbitrary rates.

2.2. Point Cloud AutoEncoder

Numerous prior studies have leveraged the autoencoder framework to achieve self-supervised representation learning for point clouds. In this approach, an encoder and a decoder are trained concurrently, where the encoder transforms the input point cloud into a latent code, and the decoder aims to reconstruct the original point cloud from the latent code. Autoencoders have been widely used for point cloud reconstruction, denoising, and generation [39].

Girdhar et al. introduced TL-Net [40], which posits that point cloud representations should exhibit generative properties in 3D space and predictability when projected into 2D space. TL-Net utilizes a 3D autoencoder to reconstruct volumetric grids in 3D space and employs a 2D convolutional network to capture 2D features from the corresponding projected images. Yang et al. innovatively devised FoldingNet [41], which incorporates a decoder based on folding principles. This decoder deforms a standard 2D grid to conform to the three-dimensional surface of the underlying object in a point cloud. Groueix et al. proposed AtlasNet [42], which represents three-dimensional shapes as a collection of parameterized surface elements. In contrast to methods that generate voxel grids or point clouds, this approach naturally infers the surface representation of the shape. L2G

Auto-encoder [43] incorporates hierarchical self-attention within the encoder to aggregate information and utilizes a recurrent neural network (RNN) as the decoder for both local and global point cloud reconstruction. Zhao et al. expanded the application of capsule networks to 3D point cloud processing, introducing a 3D capsule network that is capable of acquiring versatile representations from unorganized 3D data [44]. Chen et al. formulated a deep autoencoder that leverages graph topology inference and filtering to extract concise representations from 3D point clouds [45]. Gao et al. introduced a graph-based autoencoder capable of capturing intrinsic patterns within point-cloud structures, accommodating both global and local transformations [46]. Eckart et al. proposed a versatile method for 3D self-supervised representation learning to address the challenges of pre-training on 3D data [47]. The approach involves softly segmenting 3D points and leveraging a generative model for these partitions to maximize data likelihood, encouraging learned representations to capture rich geometric information. Pang et al. presented a novel self-supervised learning approach called Masked Autoencoders for Point Cloud, leveraging techniques successful in natural language processing and computer vision, showcasing efficiency in pre-training and outperforming other self-supervised methods on downstream tasks [48]. Zhang et al. proposed Point-M2AE [49], a novel hierarchical self-supervised learning framework utilizing Multi-scale Masked Autoencoders for effective 3D point cloud pre-training, demonstrating state-of-the-art performance in downstream tasks and surpassing fully trained methods in certain cases.

Similar to FoldingNet [41] and AtlasNet [42], RE-PU can be considered as mapping a predefined prior distribution, such as a unit square, onto a three-dimensional surface representation of a point cloud. The role of RE-PU's decoder is to reconstruct the prior distribution into the input point cloud, while the encoder primarily serves to obtain the identification of the input point cloud. If a separate model is trained for each shape, the encoder can be removed, and the decoder can be trained alone.

2.3. Implicit Neural 3D Representation

Implicit neural representation has been widely used in various fields, including computer graphics, computer vision, and robotics. It represents objects as the zero level set of a function, which can be learned from point clouds or voxels [50,51]. An implicit representation of a 3D shape involves a deep network that transforms 3D coordinates into signed distances or occupancy grids [52,53]. In contrast to explicit representations such as point clouds, voxels, or triangle meshes, implicit representations provide a continuous representation for the shape and eliminate the challenges associated with discretization errors. In the context of point cloud upsampling, implicit neural representation can be employed to generate new points consistent with the original geometry during upsampling while avoiding distortions such as noise and artifacts.

In our approach, RE-PU serves as a neural parametric surface and can also be conceptualized as a constrained Distance Function. This is because our input involves sampled distributions rather than three-dimensional space, and the output consists of three-dimensional coordinates instead of real numbers. Therefore, RE-PU can be regarded as obtaining a continuous representation from the discrete representation of point clouds, allowing us to achieve upsampling at arbitrary rates.

3. Method

3.1. Problem Formulation

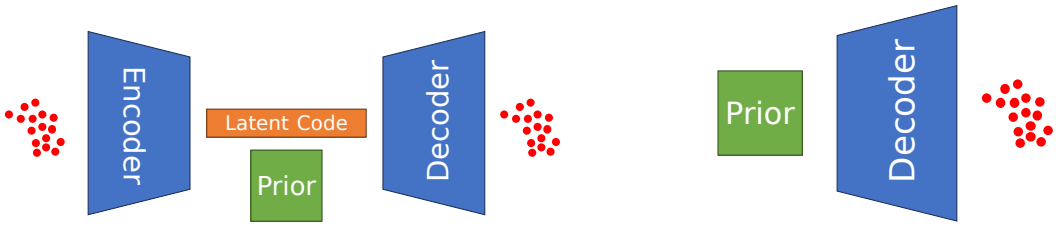
Given a sparse point cloud $X = \{x_1, x_2, \cdots, x_n\}$, where $x_i \in \mathbb{R}^3$, the point cloud upsampling problem aims to generate a dense point cloud $Y = \{y_1, y_2, \cdots, y_m\}$, where $m > n$. X is not necessarily a subset of Y, and the points in Y are not required to be in the same order as the points in X. The upsampling rate r is defined as the ratio of the number of points in the dense point cloud to the number of points in the sparse point cloud, i.e., $r = m/n$. The newly generated point cloud should be uniformly distributed on the surface represented by the original input point cloud. The point cloud upsampling

problem can be formulated as a function f that maps the sparse point cloud to the dense point cloud, i.e., $\mathrm{Y} = f(\mathrm{X})$.

3.2. Overview

We propose a novel point cloud upsampling method, named RE-PU, which is based on the point cloud reconstruction from a prior distribution, achieving self-supervised upsampling at arbitrary rates. The idea is simple but effective: we train a network to reconstruct a prior distribution into the input point cloud, and then we can upsample the point cloud data by increasing the number of sampled points on the prior distribution.

The architecture of RE-PU is illustrated in Figure 1. The network can be an AutoEncoder (Figure 1a), which is for multiple shapes in one model, consisting of an encoder and a decoder, where the encoder identifies input point clouds and the decoder takes the encoder's output along with a prior distribution to reconstruct the prior distribution into the input point cloud. The difference between RE-PU and a regular AutoEncoder lies in the decoder, which incorporates a prior input.



(**a**) AutoEncoder. (**b**) Decoder-only.

Figure 1. The architecture of RE-PU.

The proposed method consists of two main stages: the first stage is to train the autoencoder with the original point cloud data and a prior distribution. And the second stage is to upsample the point cloud data with the trained autoencoder. If a separate model is trained for a single shape, the encoder can be removed, and the decoder can be trained alone (Figure 1b).

Since the object is to reconstruct the original point cloud, we do not need input and ground truth paired data, achieving self-supervised upsampling. Since the prior distribution is continuous, then we successfully transform the discrete representation, i.e., the point cloud, into a continuous representation, so we can upsample the point cloud at arbitrary rates.

3.3. Network

A. **Encoder Based on Dynamic Graph**

The encoder of RE-PU utilizes DGCNN [11], which is a symmetric function that takes a point cloud as input and outputs a feature vector for each point in the point cloud. The encoder captures local information by adaptively adjusting the neighborhood graph in the feature space.

As shown in Figure 2, the encoder adopts a two-layer structure, where each layer constructs a local dynamic graph based on feature similarity using K-nearest neighbors (KNN), followed by convolutional operations to obtain local features. The neighborhood of a point shifts across layers due to the distinct features of each layer, leading to a dynamic graph. Thus, traversing through multiple layers allows for the acquisition of a wider spectrum of information, surpassing the constrained neighborhood of the initial points. This hierarchical convolutional structure is widely utilized in image processing and proves to be an effective method for extracting local features [54].

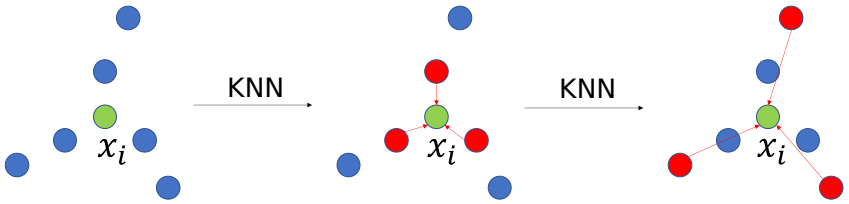


Figure 2. Expanded receptive field.

Figure 3 illustrates the specific operations of the dynamic graph at each layer in the encoder. Given a point cloud $L_i \in \mathbb{R}^{N*C_i}$ with N points in C_i dimensions as input, the output is a point cloud $L_o \in \mathbb{R}^{N*C_o}$ with N points in C_o dimensions, where C_o can be equal to C_i. Initially, for every point x_i within the point cloud, a directed graph is fashioned by gauging feature similarity via K-nearest neighbors (KNN), where each point is capable of referencing itself. The feature of each edge in the graph, linking point x_i to each point x_j in its neighborhood $N(i)$, is derived by subtracting the feature of x_i from that of x_j. Subsequently, the feature of the edge is concatenated with the self-feature of x_i, culminating in a comprehensive feature representation that encapsulates both local and global information from the neighborhood graph. Following this, this feature undergoes convolutional layer processing, followed by max aggregation across the neighborhood, ultimately yielding the feature x_i' for the point x_i, as depicted in Equation (1).

$$x_i' = \max_{j \in N(i)} \left(Conv(x_j - x_i) \oplus x_i \right) \tag{1}$$

where $\oplus$ signifies the concatenation operation, and $Conv$ denotes the convolutional operation.

Following two layers of convolutional operations, the feature of each point is refined by leveraging the features of its surrounding neighborhood. Similar to PointNet [6], a max pooling operation is applied to the entire point cloud to derive its global feature, which is then fed into the decoder for further processing.

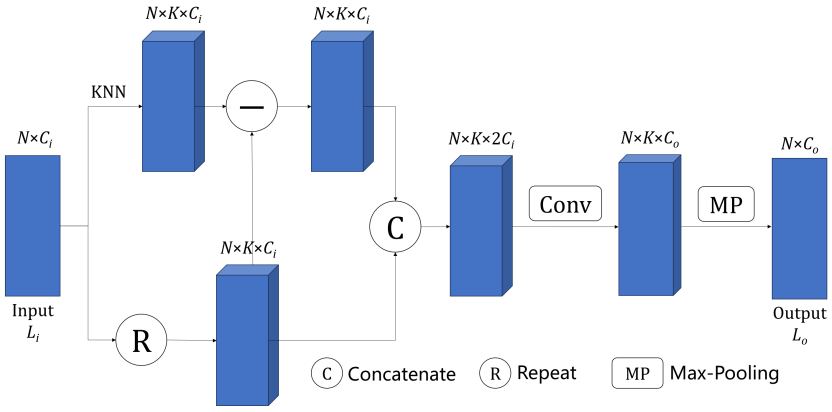


Figure 3. Details of dynamic graph.

B. Decoder based on Offset Attention

The inherent permutation invariance of attention mechanisms makes them naturally well-suited for point cloud processing and enables effective handling of the issue of long-range encoding. Therefore, they are widely applied in point cloud processing [55]. Unlike typical self-attention mechanisms, the decoder in this paper employs Laplacian-based offset attention [12]. This approach enables the network to achieve a more holistic understanding of the point cloud and enhances the preservation of point cloud details.

We concatenate the output of the decoder with the sampled points from the prior distribution and use it as the input to the decoder. Illustrated in Figure 4, input a point cloud $G_i \in \mathbb{R}^{N*D_i}$ with N points in D_i dimensions. The queries (Q), keys (K), and values

(V) are obtained through linear transformations. Attention weights are obtained through the following formula:

$$\widetilde{A} = \widetilde{a}_{ij} = Q^T \cdot K \tag{2}$$

$$\overline{a}_{ij} = softmax(\widetilde{a}_{ij}) = \frac{\exp(\widetilde{a}_{ij})}{\sum_k \exp(\widetilde{a}_{kj})} \tag{3}$$

$$a_{ij} = \frac{\overline{a}_{ij}}{\sum_k \overline{a}_{ik}} \tag{4}$$

The process involves matrix multiplication of Q and K, followed by applying the softmax operation along the first dimension and using L1 Norm operation along the second dimension to obtain normalized attention weights $A = (a_{ij})$. This approach sharpens the weights and diminishes the impact of noise. Afterwards, the output features are obtained through matrix multiplication between A and V, and finally, the output point cloud is generated according to Equation (6).

$$G_{sa} = A \cdot V \tag{5}$$

$$G_o = MLP(G_i - G_{sa}) + G_i \tag{6}$$

where G_{sa} denotes the self-attention output of the point cloud, and MLP represents a multi-layer perceptron. The difference $G_i - G_{sa}$ can be analogized to the discrete Laplacian operator, and graph convolutional networks have demonstrated the advantage of the Laplacian matrix over the adjacency matrix.

The decoder consists of 4 layers of offset attention, where the output of each layer serves as the input for the next layer. Finally, the model is regressed back to the Euclidean space of the point cloud via a fully connected layer.

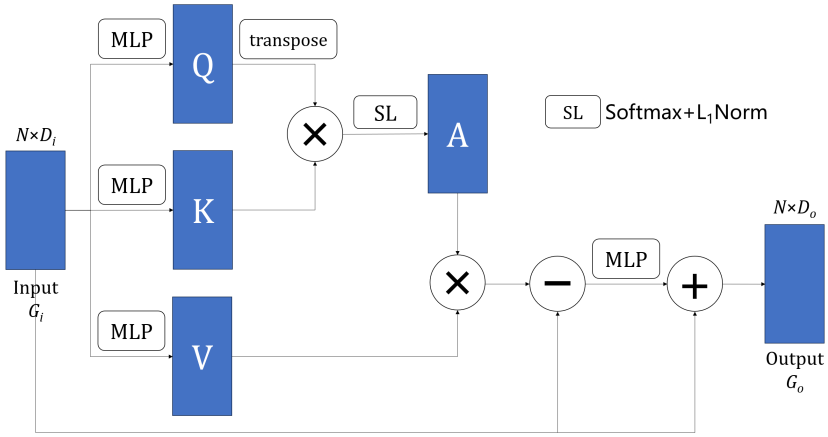
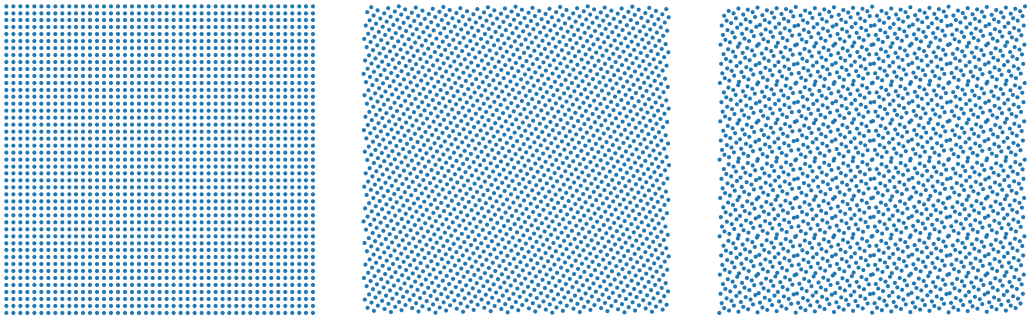


Figure 4. Details of Offset Attention.

C. Prior Distribution

The prior distribution is a continuous distribution from which we sample points to reconstruct the original point cloud. The prior distribution can be a unit square, unit sphere, and so on. In Figure 5, the unit square is taken as an example. We input the sampled points from the prior distribution, alongwith the latent code of the encoder, into the decoder to reconstruct the original point cloud. The sampled points from the prior distribution can be lattice points (Figure 5a), Fibonacci lattice (Figure 5b), Hammersley points (Figure 5c), and so on. And the additional noise can be added to the sampled points during the training process. Through a progressive training approach, we continuously adjust the number of sampled points in each reconstruction process, gradually converging to the shape represented by the input point cloud. It can be considered that the decoder transforms this prior distribution into the shape of the three-dimensional surface represented by the point cloud. In this study, we conducted experiments with various prior distributions.



(**a**) Lattice points. (**b**) Fibonacci points. (**c**) Hammersley points

Figure 5. Different prior distributions.

D. Loss Function

The training process minimizes the reconstruction loss between the reconstructed point cloud and the original point cloud. We utilize the Chamfer distance as the reconstruction loss, which is defined as the average distance from each point in the reconstructed point cloud to the nearest point in the original point cloud, and vice versa.

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\min_{j=1}^{N}||x_i - x'_j||_2^2 + \frac{1}{N}\sum_{j=1}^{N}\min_{i=1}^{N}||x_i - x'_j||_2^2 \tag{7}$$

where x_i represents the i-th point in the original point cloud, x'_j represents the j-th point in the reconstructed point cloud, N is the number of points in the original point cloud and the reconstructed point cloud, and $||\cdot||_2^2$ denotes the squared Euclidean distance.

3.4. Point Cloud Reconstruction and Upsampling

For models handling multiple shapes, we trained an autoencoder. For single shapes, we utilized the decoder-only architecture. The entire training process is end-to-end, with the training data consisting of discrete sampled points. As shown in Figure 6, we adopted a progressive training approach, continuously adjusting the number of sampled points throughout the training process. Additionally, we introduced noise to the sampled points during the training process to ensure coverage of the entire prior distribution.

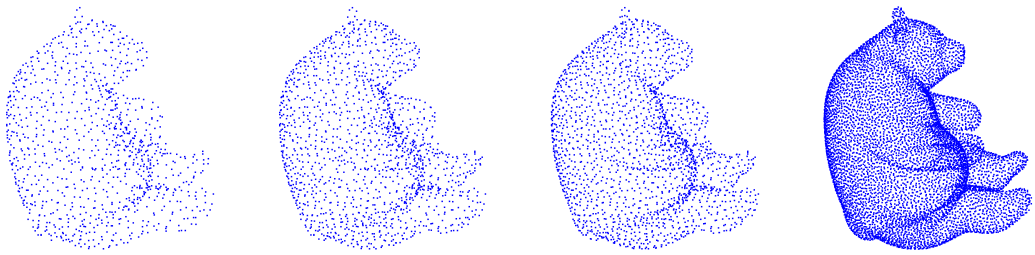


Figure 6. Progressive reconstruction and upsampling. From left to right: reconstruction of 1024 points, 1536 points, 2048 points, and upsampling of 8192 points.

After the training process, we could consider whether a continuous representation had been obtained from the discrete representation of the point cloud. The decoder can be used alone for point cloud upsampling, and the number of sampled points from the prior distribution can be adjusted to achieve the desired upsampling rate. Therefore, the proposed method accomplishes self-supervised upsampling at arbitrary rates.

4. Experiments

4.1. Implementation Details

Our experiments were performed on an NVIDIA Geforce RTX 3090 GPU with 24 GB of memory, an Intel Core i7-11700K processor clocked at 3.6 GHz, and 32 GB of RAM. The implementation was carried out using PyTorch 2.0, with the Adam optimizer, a learning rate of 0.001, and a batch size of 64. The training lasted for 200 epochs. The encoder consisted of two dynamic layers with k set to 20, while the decoder comprised four offset attention layers. The upsampling ratio r was set to 4 by default, the size of the point cloud in the reconstruction task was 2048, and the size of the dense point cloud obtained in the upsampling task was 8192.

4.2. Datasets and Metrics

The experiments in this study were performed and tested on the dataset proposed in [25] (for convenience of description, denoted as PU-GAN) and the dataset PU1K proposed in [17]. The point cloud data were obtained from the surface of original 3D triangle meshes by Possion Disk Sampling. Since our method is self-supervised, we only utilized the testing set of the dataset during training, rather than the training set.

PU-GAN: The PU-GAN [25] dataset consists of a total of 147 models, with 120 designated for training and 27 for testing. This dataset is widely employed in point cloud upsampling tasks. The training data had 40 simple models, 40 medium models, and 40 complex models, covering point clouds of diverse complexities. During the training phase, each model contained 200 point clouds patches, resulting in a total of 24,000 point cloud pairs.

PU1K: The PU1K [17] dataset is currently the largest dataset in point cloud upsampling, approximately 8 times larger than the PU-GAN dataset, of which the PU-GAN dataset is a subset. The PU1K dataset comprises 1020 training models and 127 test models, with the majority sourced from ShapeNetCore [56]. During the training phase, each model contains 50 point clouds patches, resulting in a total of 69,000 point cloud pairs.

Evaluation Metrics: We utilized five quantitative evaluation metrics to evaluate the experimental results: (i) CD (Chamfer Distance, describing the average distance of the closest points of two point clouds); (ii) HD (Hausdorff Distance, describing the farthest distance between the closest points of two point clouds); (iii) EMD (Earth Mover Distance, describing the average distance of optimal transportation between two point clouds); (iv) P2F (Point-to-Surface Distance, describing the distance from the generated point to the original surface). Among them, P2F includes the mean and variance, representing its mathematical expectation and degree of dispersion, respectively. For all five metrics, smaller values indicate better performance.

4.3. Quantitative and Qualitative Results

This section provides a quantitative and qualitative comparison with other methods on the PU-GAN [25] dataset and PU1K [17] dataset. The compared methods include PU-Net [14], MPU [16], PU-GAN [25], PU-GCN [17], Dis-PU [18], SSAS [31], and Grad-PU [38], where SSAS [31] is an unsupervised method, and the others are supervised methods.

PU-GAN: We retrained the PU-Net [14], MPU [16], and SSAS [31] models on the PU-GAN [25] dataset. The remaining models, PU-GAN [25], PU-GCN [17], Dis-PU [18], and Grad-PU [38], utilized the pre-trained models released by the authors along with the data mentioned in the paper. Table 1 illustrates the quantitative comparison between RE-PU and other methods, where bold denotes the best performance, and underline denotes the second-best. Our input point cloud size is fixed at 2048 points. We achieved optimal results in both CD and EMD metrics and the second-best results in HD and P2F (std) metrics. Figure 7 illustrates the qualitative comparison between RE-PU and other methods. As seen in the first row showcasing the bird's claws, RE-PU exhibits finer details, with sharp features closely resembling the ground truth. In the second and third rows showcasing enlarged portions of fingers and models, RE-PU generates fewer noises and outliers, thereby

preserving the original structure of the point cloud and closely approaching the ground truth in more intricate regions.

Table 1. Quantitative results on PU-GAN [25] dataset.

Method	CD ↓	HD ↓	EMD ↓	P2F (avg) ↓	P2F (std) ↓
PU-Net [14]	0.556	4.750	40.146	4.678	5.946
MPU [16]	0.298	4.700	30.534	2.855	5.180
PU-GAN [25]	0.280	4.640	26.243	2.330	4.431
PU-GCN [17]	0.258	**1.885**	24.460	2.721	3.542
Dis-PU [18]	0.260	2.104	25.312	2.480	3.521
SSAS [31]	0.264	2.320	25.027	2.625	3.462
Grad-PU [38]	0.245	2.369	23.348	**1.893**	**2.875**
Ours	**0.238**	2.012	**22.353**	2.463	2.965

Note: The downward arrows (↓) indicate that lower values are better for the corresponding metrics.

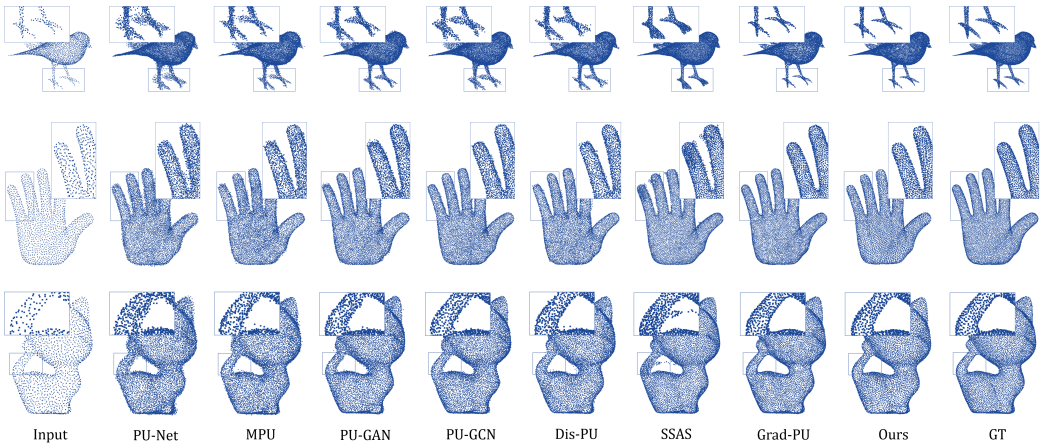


Figure 7. Qualitative results on PU-GAN [25] dataset.

PU1K: We retrained the models of PU-GAN [25], Dis-PU [18], and SSAS [31] on the PU1K [17] dataset, while the rest utilized the pre-trained models and data provided by the authors for PU-GCN [17] and Grad-PU [38]. Table 2 presents the quantitative comparison between RE-PU and other methods. Similar to the previous table, bold denotes the best performance, and italic indicates the second-best. The size of the input point cloud is also 2048. It can be observed that, although Grad-PU [38] performs well in CD and P2F (avg) metrics, RE-PU achieves the best results in other metrics, attaining the second-best results in the remaining two, with the output results closest to the target point cloud. Qualitative comparisons are depicted in Figure 8, where in the first and second rows, RE-PU exhibits fewer artifacts and better preserves the original structure of the point cloud. In the third row featuring the hat model, it is evident that RE-PU effectively preserves the original structure represented by the point cloud, while other methods generate more outliers and even attempt to merge different parts of the original point cloud.

Table 2. Quantitative results on PU1K [17] dataset.

Method	CD ↓	HD ↓	EMD ↓	P2F (avg) ↓	P2F (std) ↓
PU-Net [14]	1.155	15.170	91.487	4.834	6.799
MPU [16]	0.935	13.327	77.401	3.551	5.970
PU-GAN [25]	0.873	12.146	68.534	3.189	5.682
PU-GCN [17]	0.585	7.577	55.570	2.499	4.004
Dis-PU [18]	0.541	8.348	53.687	2.964	5.209
SSAS [31]	0.613	7.451	68.970	2.474	6.088
Grad-PU [38]	**0.403**	3.743	55.487	**1.480**	2.468
Ours	0.421	**3.236**	**46.476**	2.257	**2.375**

Note: The downward arrows (↓) indicate that lower values are better for the corresponding metrics.

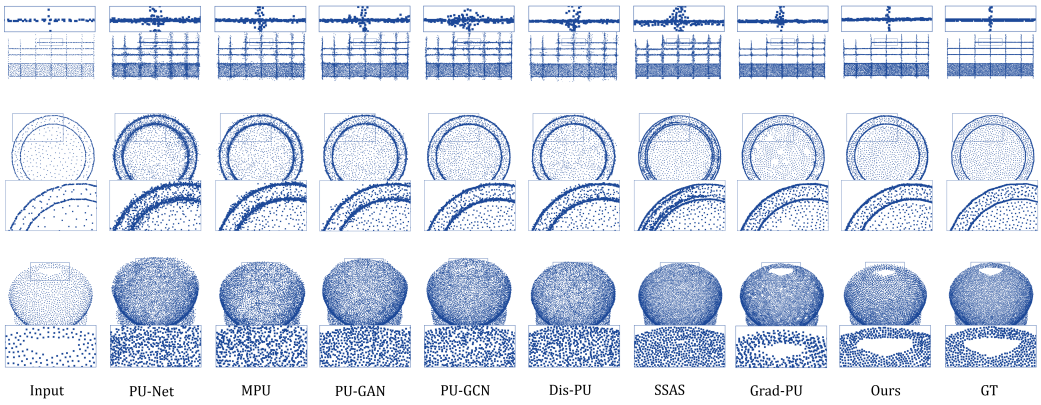


Figure 8. Qualitative results on PU1K [17] dataset.

Overall, our results were highly competitive, demonstrating the robustness and reliability of our approach across different evaluation metrics. This indicates the effectiveness of our method in capturing both global and local features, thereby enhancing its applicability across various point cloud processing tasks. Although we do not have dense point clouds corresponding to sparse point clouds as labels, we still achieved comparable results to the state-of-the-art supervised methods. Furthermore, compared to unsupervised methods, our approach exhibits a more significant improvement.

4.4. Other Experiments

A. **Results of Different Sizes of Point Clouds**

Figure 9 illustrates the results obtained when the same model is provided with inputs of varying resolutions. Even with just 256 input points, RE-PU maintains the original shape of the point cloud remarkably well, exhibiting minimal generation of outliers. This indicates the robustness of our approach across different input resolutions, highlighting its ability to effectively handle point clouds of varying densities while maintaining fidelity to the original shape. Such versatility enhances its applicability in real-world scenarios where input point cloud resolutions may vary.

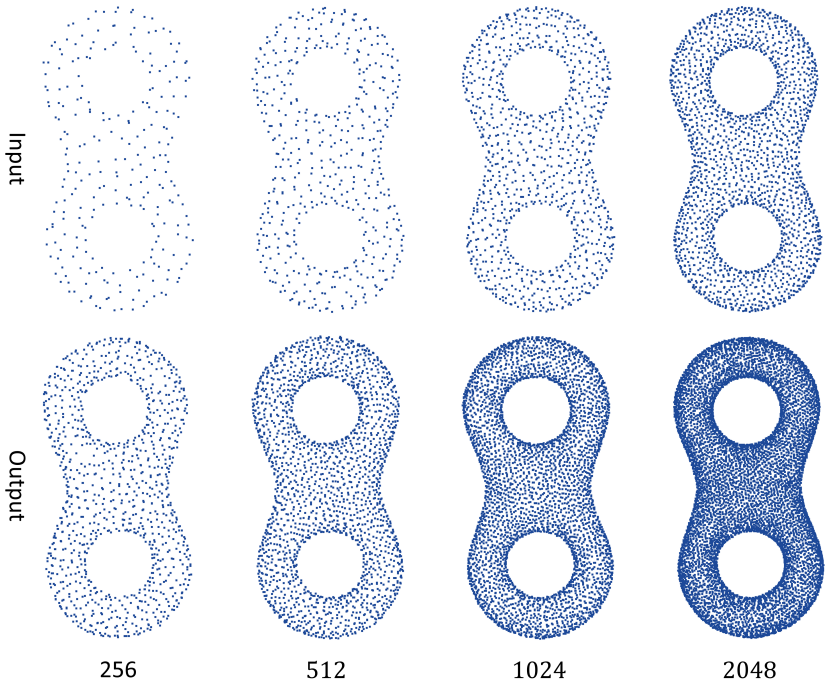


Figure 9. Qualitative results with varying sizes.

B. Results of Noisy Point Clouds

Figure 10 illustrates the comparative results under different noise levels. As the noise level increases, RE-PU continues to strive to preserve the original shape of the point cloud, while PU-GAN and PU-GCN generate more outliers, leading to a compromise in the fidelity of the original shape. This demonstrates that even in the presence of noise, our method outperforms others. Certainly, as the level of noise increases, the quality of upsampling inevitably diminishes.

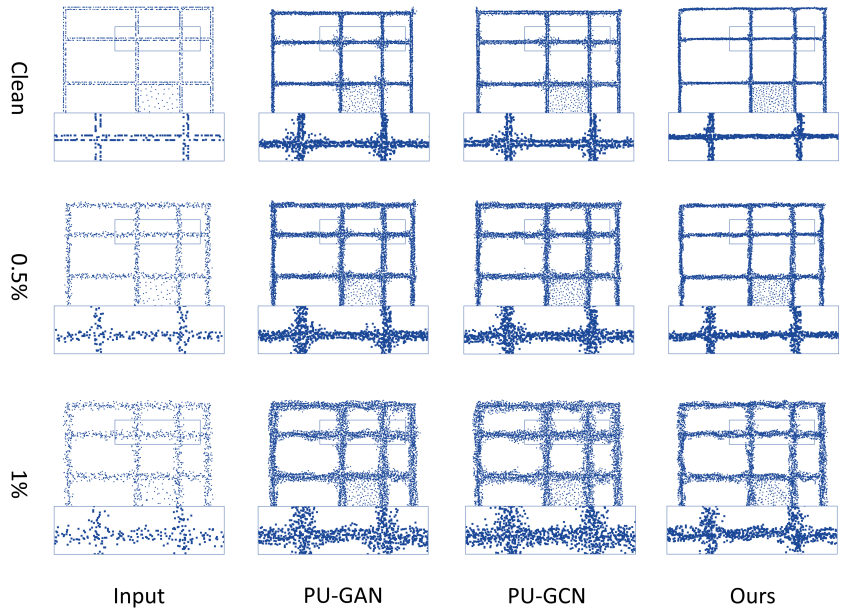


Figure 10. Qualitative results with varying noise levels.

C. Results of Varying Upsampling Rates

As previously discussed, RE-PU offers the capability to upsample point clouds at arbitrary rates, providing flexibility in adapting to different resolution requirements. Figure 11 showcases the outcomes achieved by the same model under various upsampling rates. RE-PU consistently delivers high-quality upsampled point clouds that closely resemble the original data, regardless of the chosen upsampling rate. This versatility underscores the adaptability and effectiveness of RE-PU in handling diverse point cloud processing tasks.

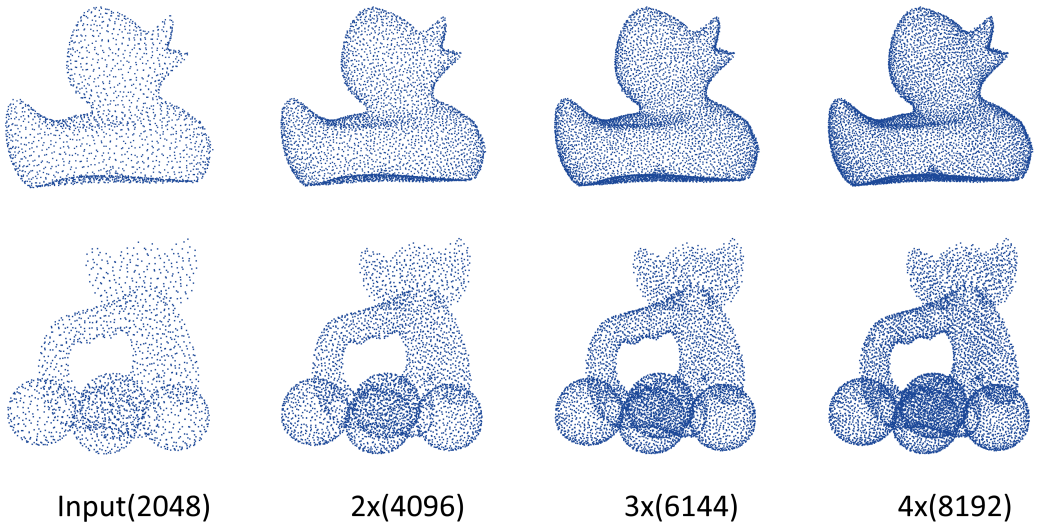


Figure 11. Qualitative results with varying upsampling rates.

4.5. Analysis

A. Reconstruction and Upsampling

Our point cloud upsampling method RE-PU is based on point cloud reconstruction. In this approach, point cloud reconstruction serves as the cornerstone of point cloud upsampling, with the quality of reconstruction directly influencing the upsampling outcomes. Figure 12 illustrates the results of point cloud reconstruction and upsampling. The x-axis represents different times in one experiment, and the y-axis represents the Chamfer distance. The reconstructed point cloud has a size of 2048, while the upsampled point cloud has a size of 8192. It is evident from the results that the quality of the upsampling outcome is intricately linked to the accuracy and fidelity of the reconstruction process. A higher quality reconstruction leads to superior upsampling results, highlighting the critical role of reconstruction in the overall effectiveness of our approach.

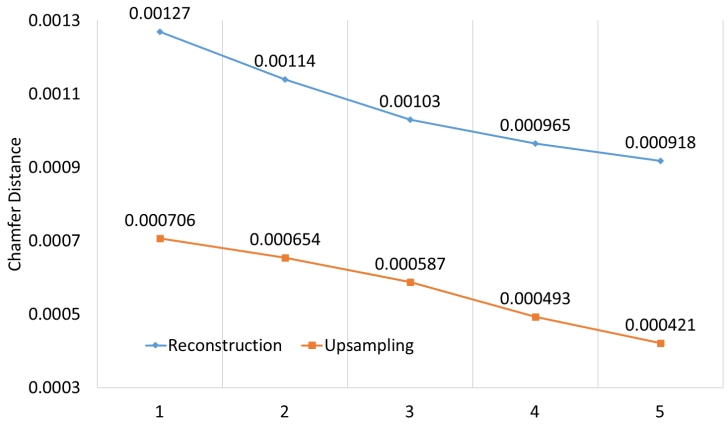


Figure 12. Reconstruction and upsampling.

B. Prior Distribution

We conducted an ablation study to investigate the effect of the prior distribution on the performance of the proposed method. The experimental findings underscore the pivotal role of the prior distribution in augmenting both the visual fidelity and quantitative performance of our approach. Table 3 illustrates the quantitative comparison across different prior distributions, where bold denotes the best performance. Specifically, "Sphere Uniform" refers to a uniform distribution on the surface of a unit sphere, and "Lattice Points + Noise" represents Lattice Points with additional noise. Interestingly, the results exhibit minimal deviation across different sampling methodologies on the unit square. However, the introduction of supplementary noise to the sampling points markedly enhances performance. Conversely, outcomes from random distributions falter, potentially attributable to challenges in achieving convergence.

Table 3. Quantitative results of different prior distributions.

Prior Distribution	Lattice Points	Fibonacci Lattice	Hammersley Points	Sphere Uniform	Lattice Points + Noise
CD ↓	0.510	0.496	0.507	0.613	0.421

Note: The downward arrows (↓) indicate that lower values are better for the corresponding metrics.

C. Encoder and Decoder

This section aims to validate the effectiveness of dynamic graphs and offset attention. The architecture of Model 1 remains consistent with RE-PU but utilizes static graphs based on point cloud coordinate space in the encoder and employs a self-attention mechanism

in the decoder, serving as the baseline model. Building upon this foundation, Model 2 is derived by replacing static graphs with dynamic graphs, while Model 3 is created by substituting self-attention with offset attention. The complete RE-PU model is obtained by combining dynamic graphs and offset attention. The quantitative results for different models are shown in Table 4. The RE-PU network, incorporating dynamic graphs and offset attention modules, delivers the most promising outcomes, showcasing a considerable enhancement compared to Model 1. Moreover, Model 3 shows a more significant improvement compared to Model 2, implying that offset attention may play a more pivotal role than dynamic graphs. Additionally, it suggests that the decoder might have a greater impact on the overall network performance than the encoder.

Table 4. Quantitative results of different models.

Model	Model 1	Model 2	Model 3	Ours
CD↓	0.874	0.697	0.512	**0.421**

Note: The downward arrows (↓) indicate that lower values are better for the corresponding metrics.

The visualization results in Figure 13 further corroborate that the model incorporating all modules generates point clouds that closely resemble the ground truth. From the bottom of the package in the first row and the shape in the second row, it can be observed that Model 1, Model 2, and Model 3 produce a significant amount of noise and outliers.

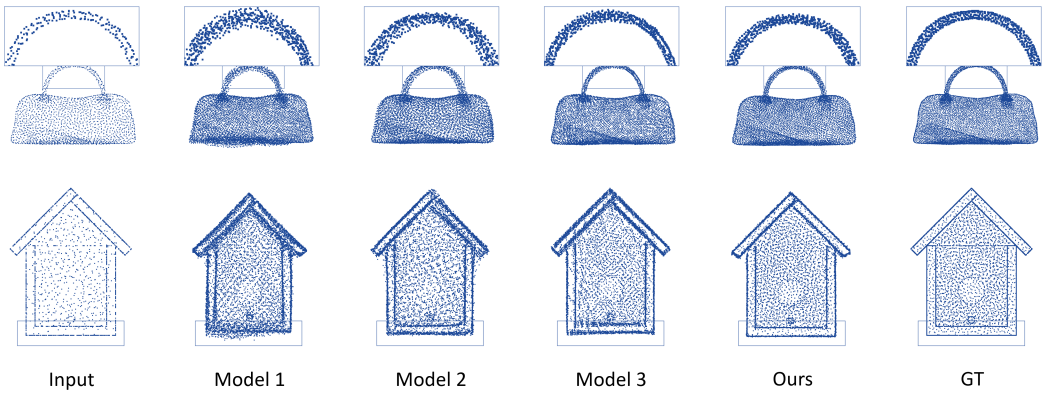


Figure 13. Qualitative results of different models.

We further explore the impact of the neighborhood size on the dynamic graph. Table 5 illustrates the quantitative comparison across different neighborhood sizes, where bold denotes the best performance. The results demonstrate that the optimal neighborhood size is 20, which aligns with the default setting in our model. Both an overly large or overly small neighborhood can lead to poor results. A neighborhood that is too small will result in insufficient neighborhood information, while a neighborhood that is too large will include more irrelevant information.

Table 5. Effect of neighborhood size on dynamic graph.

KNN	10	15	20	25	30
CD ↓	0.598	0.523	**0.421**	0.498	0.592

5. Conclusions

In this paper, we propose RE-PU, a novel point cloud upsampling method, which is based on point cloud reconstruction, achieving self-supervised upsampling at arbitrary rates. This feature can expand the scope of application of the method.

The proposed method consists of two main stages: the first stage involves training the autoencoder or decoder using the original point cloud data and a prior distribution. The

encoder identifies input point clouds, and the decoder takes the encoder's output along with a prior distribution as input to reconstruct the prior distribution into the input point cloud. And the second stage is to upsample the point cloud data by increasing the number of sampled points on the prior distribution with the trained model. The experimental results demonstrate that the proposed method can achieve comparable outcomes to other state-of-the-art methods in terms of both visual quality and quantitative metrics. We achieved the best results in two indicators and second-best results in the remaining two on the PU-GAN dataset. Our results show a 4% enhancement in both the CD and EMD metrics over the current state-of-the-art methods. We also achieved the best results in three indicators and second-best results in the remaining two on the PU1K dataset. We improved the existing best results by about 13% on the HD metric, 16% on the EMD metric, and 4% on the P2F (std) metric.

Additionally, we explored the effectiveness of our network architecture and the impact of different prior distributions. It turns out that the dynamic graph and offset attention modules play a crucial role in the network, which can significantly improve the performance of the network compared to the baseline model. The results also show that the choice of prior distribution can have a significant impact on the performance of the network, and the introduction of noise to the sampled points can enhance the performance of the network.

We also demonstrate the results of our method on point clouds of different scenes. The results show that our method can effectively handle noisy point clouds and point clouds of different sizes, and it can upsample point clouds at arbitrary rates.

In the future, we will delve deeper into exploring the relationship between point cloud reconstruction and point cloud upsampling. Additionally, we plan to investigate the application of the proposed method in other tasks, such as point cloud completion and point cloud generation.

Author Contributions: Conceptualization, Y.H.; methodology, M.Y.; validation, F.Z.; formal analysis, F.Y.; writing—original draft preparation, Y.H.; writing—review and editing, M.Y.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Luo, L.; Tang, L.; Zhou, W.; Wang, S.; Yang, Z.X. Pu-eva: An edge-vector based approximation solution for flexible-scale point cloud upsampling. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 16208–16217.
2. Liu, Y.; Wang, Y.; Liu, Y. Refine-PU: A Graph Convolutional Point Cloud Upsampling Network using Spatial Refinement. In Proceedings of the 2022 IEEE International Conference on Visual Communications and Image Processing (VCIP), Suzhou, China, 13–16 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–5.
3. Li, T.; Lin, Y.; Cheng, B.; Ai, G.; Yang, J.; Fang, L. PU-CTG: A Point Cloud Upsampling Network Using Transformer Fusion and GRU Correction. *Remote. Sens.* **2024**, *16*, 450. [CrossRef]
4. Akhtar, A.; Li, Z.; Van der Auwera, G.; Li, L.; Chen, J. Pu-dense: Sparse tensor-based point cloud geometry upsampling. *IEEE Trans. Image Process.* **2022**, *31*, 4133–4148. [CrossRef] [PubMed]
5. Huang, H.; Wu, S.; Gong, M.; Cohen-Or, D.; Ascher, U.; Zhang, H. Edge-aware point set resampling. *ACM Trans. Graph. (TOG)* **2013**, *32*, 1–12. [CrossRef]
6. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
7. Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; 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, Honolulu, HI, USA, 21–26 July 2017; pp. 4681–4690.

8. Du, H.; Yan, X.; Wang, J.; Xie, D.; Pu, S. Point cloud upsampling via cascaded refinement network. In Proceedings of the Asian Conference on Computer Vision, Macao, China, 4–8 December 2022; pp. 586–601.
9. Qiu, S.; Anwar, S.; Barnes, N. Pu-transformer: Point cloud upsampling transformer. In Proceedings of the Asian Conference on Computer Vision, Macao, China, 4–8 December 2022; pp. 2475–2493.
10. Lim, S.; El-Basyouny, K.; Yang, Y.H. PU-Ray: Domain-Independent Point Cloud Upsampling via Ray Marching on Neural Implicit Surface. *IEEE Trans. Intell. Transp. Syst.* **2024**, *1*, 1–11. [CrossRef]
11. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph.* **2019**, *38*, 1–12. [CrossRef]
12. Guo, M.H.; Cai, J.X.; Liu, Z.N.; Mu, T.J.; Martin, R.R.; Hu, S.M. Pct: Point cloud transformer. *Comput. Vis. Media* **2021**, *7*, 187–199. [CrossRef]
13. Zhang, Y.; Zhao, W.; Sun, B.; Zhang, Y.; Wen, W. Point cloud upsampling algorithm: A systematic review. *Algorithms* **2022**, *15*, 124. [CrossRef]
14. Yu, L.; Li, X.; Fu, C.W.; Cohen-Or, D.; Heng, P.A. Pu-net: Point cloud upsampling network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2790–2799.
15. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5105–5114.
16. Yifan, W.; Wu, S.; Huang, H.; Cohen-Or, D.; Sorkine-Hornung, O. Patch-based progressive 3D point set upsampling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5958–5967.
17. Qian, G.; Abualshour, A.; Li, G.; Thabet, A.; Ghanem, B. Pu-gcn: Point cloud upsampling using graph convolutional networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 11683–11692.
18. Li, R.; Li, X.; Heng, P.A.; Fu, C.W. Point cloud upsampling via disentangled refinement. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 344–353.
19. Long, C.; Zhang, W.; Li, R.; Wang, H.; Dong, Z.; Yang, B. Pc2-pu: Patch correlation and point correlation for effective point cloud upsampling. In Proceedings of the 30th ACM International Conference on Multimedia, Lisboa, Portugal, 10–14 October 2022; pp. 2191–2201.
20. Wang, J.; Chen, J.; Shi, Y.; Ling, N.; Yin, B. SSPU-Net: A Structure Sensitive Point Cloud Upsampling Network with Multi-Scale Spatial Refinement. In Proceedings of the 31st ACM International Conference on Multimedia, Ottawa, ON, Canada, 29 October–3 November 2023; pp. 1546–1555.
21. Zhao, W.; Zhang, H.; Zheng, C.; Yan, X.; Cui, S.; Li, Z. CPU: Codebook Lookup Transformer with Knowledge Distillation for Point Cloud Upsampling. In Proceedings of the 31st ACM International Conference on Multimedia, Ottawa, ON, Canada, 29 October–3 November 2023; pp. 3917–3925.
22. Cai, P.; Wu, Z.; Wu, X.; Wang, S. Parametric Surface Constrained Upsampler Network for Point Cloud. *arXiv* **2023**, arXiv:2303.08240.
23. Qian, Y.; Hou, J.; Kwong, S.; He, Y. PUGeo-Net: A geometry-centric network for 3D point cloud upsampling. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Cham, Switzerland, 2020; pp. 752–769.
24. Qian, Y.; Hou, J.; Kwong, S.; He, Y. Deep magnification-flexible upsampling over 3d point clouds. *IEEE Trans. Image Process.* **2021**, *30*, 8354–8367. [CrossRef] [PubMed]
25. Li, R.; Li, X.; Fu, C.W.; Cohen-Or, D.; Heng, P.A. Pu-gan: A point cloud upsampling adversarial network. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 21 October–2 November 2019; pp. 7203–7212.
26. Liu, H.; Yuan, H.; Hou, J.; Hamzaoui, R.; Gao, W. Pufa-gan: A frequency-aware generative adversarial network for 3d point cloud upsampling. *IEEE Trans. Image Process.* **2022**, *31*, 7389–7402. [CrossRef] [PubMed]
27. Zhou, K.; Dong, M.; Arslanturk, S. "Zero-Shot" Point Cloud Upsampling. In Proceedings of the 2022 IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan, 18–22 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–6.
28. Kumbar, A.; Anvekar, T.; Tabib, R.A.; Mudenagudi, U. ASUR3D: Arbitrary Scale Upsampling and Refinement of 3D Point Clouds using Local Occupancy Fields. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Paris, France, 2–6 October 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1644–1653.
29. Kumbar, A.; Anvekar, T.; Vikrama, T.A.; Tabib, R.A.; Mudenagudi, U. TP-NoDe: Topology-aware Progressive Noising and Denoising of Point Clouds towards Upsampling. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 2272–2282.
30. Zhao, Y.; Hui, L.; Xie, J. Sspu-net: Self-supervised point cloud upsampling via differentiable rendering. In Proceedings of the 29th ACM International Conference on Multimedia, Virtual, 20–24 October 2021; pp. 2214–2223.
31. Zhao, W.; Liu, X.; Zhong, Z.; Jiang, J.; Gao, W.; Li, G.; Ji, X. Self-supervised arbitrary-scale point clouds upsampling via implicit neural representation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 1999–2007.
32. Liu, X.; Liu, X.; Liu, Y.S.; Han, Z. Spu-net: Self-supervised point cloud upsampling by coarse-to-fine reconstruction with self-projection optimization. *IEEE Trans. Image Process.* **2022**, *31*, 4213–4226. [CrossRef] [PubMed]

33. Hu, X.; Mu, H.; Zhang, X.; Wang, Z.; Tan, T.; Sun, J. Meta-SR: A magnification-arbitrary network for super-resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 1575–1584.
34. Ye, S.; Chen, D.; Han, S.; Wan, Z.; Liao, J. Meta-PU: An arbitrary-scale upsampling network for point cloud. *IEEE Trans. Vis. Comput. Graph.* **2021**, *28*, 3206–3218. [CrossRef] [PubMed]
35. Feng, W.; Li, J.; Cai, H.; Luo, X.; Zhang, J. Neural points: Point cloud representation with neural fields for arbitrary upsampling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 18633–18642.
36. Mao, A.; Du, Z.; Hou, J.; Duan, Y.; Liu, Y.j.; He, Y. PU-Flow: A point cloud upsampling network with normalizing flows. *IEEE Trans. Vis. Comput. Graph.* **2022**, *29*, 4964–4977. [CrossRef] [PubMed]
37. Mao, A.; Duan, Y.; Wen, Y.H.; Du, Z.; Cai, H.; Liu, Y.J. Invertible residual neural networks with conditional injector and interpolator for point cloud upsampling. In Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, Macao, China, 19–25 August 2023; pp. 1267–1275.
38. He, Y.; Tang, D.; Zhang, Y.; Xue, X.; Fu, Y. Grad-PU: Arbitrary-Scale Point Cloud Upsampling via Gradient Descent with Learned Distance Functions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 5354–5363.
39. Xiao, A.; Huang, J.; Guan, D.; Zhang, X.; Lu, S.; Shao, L. Unsupervised point cloud representation learning with deep neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 11321–11339. [CrossRef] [PubMed]
40. Girdhar, R.; Fouhey, D.F.; Rodriguez, M.; Gupta, A. Learning a predictable and generative vector representation for objects. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part VI 14; Springer: Cham, Switzerland, 2016; pp. 484–499.
41. Yang, Y.; Feng, C.; Shen, Y.; Tian, D. Foldingnet: Point cloud auto-encoder via deep grid deformation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 206–215.
42. Groueix, T.; Fisher, M.; Kim, V.G.; Russell, B.C.; Aubry, M. A papier-mâché approach to learning 3D surface generation. In Proceedings of the IEEE Conference On Computer Vision And Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 216–224.
43. Liu, X.; Han, Z.; Wen, X.; Liu, Y.S.; Zwicker, M. L2g auto-encoder: Understanding point clouds by local-to-global reconstruction with hierarchical self-attention. In Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 21–25 October 2019; pp. 989–997.
44. Zhao, Y.; Birdal, T.; Deng, H.; Tombari, F. 3D point capsule networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 1009–1018.
45. Chen, S.; Duan, C.; Yang, Y.; Li, D.; Feng, C.; Tian, D. Deep unsupervised learning of 3D point clouds via graph topology inference and filtering. *IEEE Trans. Image Process.* **2019**, *29*, 3183–3198. [CrossRef] [PubMed]
46. Gao, X.; Hu, W.; Qi, G.J. Graphter: Unsupervised learning of graph transformation equivariant representations via auto-encoding node-wise transformations. In Proceedings of theIEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 7163–7172.
47. Eckart, B.; Yuan, W.; Liu, C.; Kautz, J. Self-supervised learning on 3D point clouds by learning discrete generative models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 8248–8257.
48. Pang, Y.; Wang, W.; Tay, F.E.; Liu, W.; Tian, Y.; Yuan, L. Masked autoencoders for point cloud self-supervised learning. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Cham, Switzerland, 2022; pp. 604–621.
49. Zhang, R.; Guo, Z.; Gao, P.; Fang, R.; Zhao, B.; Wang, D.; Qiao, Y.; Li, H. Point-m2ae: Multi-scale masked autoencoders for hierarchical point cloud pre-training. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 27061–27074.
50. Park, J.J.; Florence, P.; Straub, J.; Newcombe, R.; Lovegrove, S. Deepsdf: Learning continuous signed distance functions for shape representation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 165–174.
51. Ma, B.; Han, Z.; Liu, Y.S.; Zwicker, M. Neural-Pull: Learning Signed Distance Function from Point clouds by Learning to Pull Space onto Surface. In Proceedings of the International Conference on Machine Learning. PMLR, Virtual, 18–24 July 2021; pp. 7246–7257.
52. Chen, Z.; Zhang, H. Learning implicit fields for generative shape modeling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5939–5948.
53. Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4460–4470.
54. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, 5–9 October 2015; Proceedings, Part III 18; Springer: Cham, Switzerland, 2015; pp. 234–241.

55. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.H.; Koltun, V. Point transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 16259–16268.
56. 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.