scientific reports

Check for updates

OPEN

# Multi-stage refinement network for point cloud completion based on geodesic attention

Yuchen Chang & Kaiping Wang✉

**The attention mechanism has significantly progressed in various point cloud tasks. Benefiting from its significant competence in capturing long-range dependencies, research in point cloud completion has achieved promising results. However, the typically disordered point cloud data features complicated non-Euclidean geometric structures and exhibits unpredictable behavior. Most current attention modules are based on Euclidean or local geometry, which fails to accurately represent the intrinsic non-Euclidean characteristics of point cloud data. Thus, we propose a novel geodesic attention-based multi-stage refinement transformer network, which enables the alignment of feature dimensions among query, key, and value, and long-range geometric dependencies are captured on the manifold. Then, a novel Position Feature Extractor is designed to enhance geometric features and explicitly capture graph-based non-Euclidean properties of point cloud objects. A Recurrent Information Aggregation Unit is further applied to aggregate historical information from the previous stages and current geometric features to guide the network in the current stage. The proposed method exhibits strong competitiveness when compared to current state-of-the-art methods.**

**Keywords** Attention mechanism, Point cloud completion, Geodesic attention, Recurrent information aggregation unit

As 3D technologies, such as 3D laser scanners and depth cameras, continue to advance rapidly, point clouds have emerged as a common method for representing 3D information. The point cloud plays a key in the deep understanding of complex real-world environments and has substantially propelled progress in 3D computer vision[1], robotics[2], and augmented reality[3]. Nevertheless, the constraints of 3D scanning equipment, along with object occlusions in real-world applications, often lead to sparse and incomplete point cloud data. The missing parts can cause a significant loss of key semantic details, greatly impacting downstream tasks[4–6] including segmentation, registration, and detection. Therefore, recovering complete shapes from partial observations is significant for downstream tasks.

Point cloud completion tasks are roughly classified into traditional methods and deep learning-based algorithms. Traditional methods mainly rely on geometric and statistical techniques to optimize the model parameters[7], primarily including interpolation-based, alignment-based, and geometric-based methods. Given the slow optimization of traditional methods, recent research has favored learning-based approaches. Voxel-based techniques generally transform point clouds into 3D grids, which are then employed to complete the shape. For voxelized point clouds, it is a simple way to directly employ sophisticated CNN for shape completion [8–12]. Due to these methods involving large calculations and potential loss of geometric information, they are not considered optimal. Each point is modeled independently in point-based methods using shared MLPs, and then a global feature is aggregated using a symmetric aggregation function, maintaining the transformation invariance of the point clouds. PointNet[13] is a pioneering deep learning model designed to process 3D point clouds directly, without needing to convert the data into grids or meshes. It works by first applying a shared MLP to each individual point, which helps the model learn useful features from each point. Then, a symmetric function, such as max pooling, is used to combine the features of all points in the point cloud. This ensures that the order of the points does not affect the result. A few point-based works[14–16] are built on the PointNet due to its simplicity and adaptability. Because PointNet is unable to extract local information and neighborhood relationships effectively, PointNet++[17] introduces abstraction layers to aggregate multi-level information. PointNet++ uses the K-Nearest Neighbors algorithm to sample local neighborhoods by selecting the 'K' nearest points, forming a local region around each point to capture local geometric patterns effectively. PointNet++ then applies PointNet recursively at different scales to extract both fine-grained local features and global context. Inspired by PointNet++, PU-

Department of Computer Science, Xi'an University of Architecture and Technology, Xi'an 710055, Shaanxi Province, China. ✉email: wkp@xauat.edu.cn

Net[18] implements feature scaling through sub-pixel convolutional layers to learn multi-scale features, which aids in producing dense point clouds from initially sparse data. Moreover, since point clouds and graphs can be considered non-Euclidean structured data, the pioneering DGCNN[19] treats point clouds as graph structures and dynamically constructs a graph where the edges between points are updated during the learning process. DGCNN introduces dynamic graph convolution to learn edge features between neighboring points, enabling the model to capture more complex local and global structures. Following the DGCNN, a few graph-based completion methods[20–22] have delivered impressive outcomes by efficiently aggregating local geometric features. Additionally, prevailing point cloud completion methods[23–29] primarily focus on the framework based on encoder–decoder. The sparse and incomplete point cloud is encoded into a global feature, then decoded into a coarse shape and refined to completion. SnowflakeNet[30] adopted a multistep point generation strategy and introduced a skip-transformer that integrates spatial relationships across multiple decoding phases. Huang et al.[31] proposed a new recurrent forward framework that leverages multiple global features at successive recurrent stages to optimize computational efficiency, achieving lower memory cost and faster convergence.

Point clouds fundamentally consist of discrete 3D coordinates extracted from 3D geometries, revealing non-Euclidean properties. In summary, given the unordered and non-uniform nature of point cloud data, existing data processing methods are still unable to effectively manage the intrinsic non-Euclidean geometry[32,33] of point cloud data, making point cloud completion a challenging task. As one of the pioneering works, Yang et al.[34] proposed a general architecture FoldingNet, which utilized a folding-based decoder to reconstruct the detailed structure of point cloud objects from a canonical 2D grid with low reconstruction errors. Following FoldingNet, MSN[35] was able to predict the entire object by assessing a group of parametric surface elements. SA-Net[36] introduced a hierarchical folding-based structure-preserving decoder for shape completion. Yu et al.[26] were the first to propose a coarse-to-fine point completion framework, where a coarse shape is generated using PointNet and later refined into a detailed shape with FoldingNet. Inspired by the success of PCN, Zhang et al.[37] designed a skeleton-bridged point cloud network to complete the partial point clouds through structure estimation and surface reconstruction. Furthermore, the following method[15,16,27,30,38] especially concentrated on multi-step point cloud generation and the refining operation to recover a fine-grained detail of point cloud objects. Compared to CNNs with limited receptive fields, Transformer[39] relies on the attention mechanism's excellent capability to capture long-range relationships, achieving impressive success in the fields of NLP and CV. The attention score, derived from the similarity between query and key, is used to weight value, allowing the model to capture long-range dependencies and adapt to varying inputs. This mechanism enhances efficiency by enabling parallel processing and dynamically adjusting focus to prioritize the most important information. Recently, due to the fact that 3D point clouds are treated as a specific type of sequential data, this architecture has also been introduced into point cloud analysis tasks. Guo et al.[40] proposed PCT and Zhao et al.[41] proposed PointTransformer, respectively. They all tailored the self-attention mechanism to align with the unique properties of point cloud data, enhancing the Transformer's suitability for point cloud tasks. Yu et al.[42] developed PointTr, utilizing the transformer framework that reinterprets point cloud completion as a set-to-set translation. Wen et al.[43] introduced a transformer-enhanced representation learning network based on PMP-Net[27] and proposed PMP-Net++, significantly improving the performance of point cloud completion. FBNet[44] proposed a feedback network to refine the present features and designed a point cross transformer to build feedback connections. PointAttN[45] first eliminated the explicit local region operations and optimally utilized the advantages of self-attention and cross-attention mechanisms. Since most previous works on point cloud data only involved 3D coordinates, analyzing the geometric properties of point cloud objects is crucial for point cloud completion. However, previous works have been insufficient to accurately capture the non-Euclidean geometric properties of point cloud objects.

This study proposes a novel multi-stage refinement transformer network based on geodesic attention to extract complex non-Euclidean geometric properties in point cloud completion. Firstly, we implement global feature extraction of the partial point cloud and generate a sparse point cloud using an encoder-decoder architecture. Then, in the refinement stage, we hierarchically reconstruct the fine-grained point information of the sparse point cloud to achieve more detailed geometric shape details. The main contributions are as follows:

(1) We introduce MRNet, a novel point cloud completion network that adopts a geodesic distance score-based attention block to measure the topological attention relationships and as a result, improves the interpretation of the local geometric features represented in the point cloud.
(2) We propose a novel Position Feature Extractor, concentrating on enhancing non-Euclidean geometric properties and model spatial reasoning. We also design a Recurrent Information Aggregation Unit to aggregate previous features with current features.
(3) The results from extensive testing indicate that our method achieves competitive results on PCN, MVP, ShapeNet55/34, and KITTI datasets.

## Methods

### Network architecture

The overall network architecture of the multi-stage refinement transformer network is shown in Fig. 1. The specifics of our method are explained in detail in the subsequent section.

*Encoder*

As depicted in Fig. 1a, the Encoder is composed of three downsampling layers and two transformer layers[41] for feature extraction. The downsampling layers obtain downsampled point clouds $\{\mathcal{P}_i\}_{i=1}^{3}$ and point features $\{\ \mathrm{f}_i\}_{i=1}^{3}$ through farthest point sampling (FPS), K-Nearest Neighbors (KNN), convolution layer, and Maxpooling layer. The transformer layers are further applied to capture interim features. Given a partial point

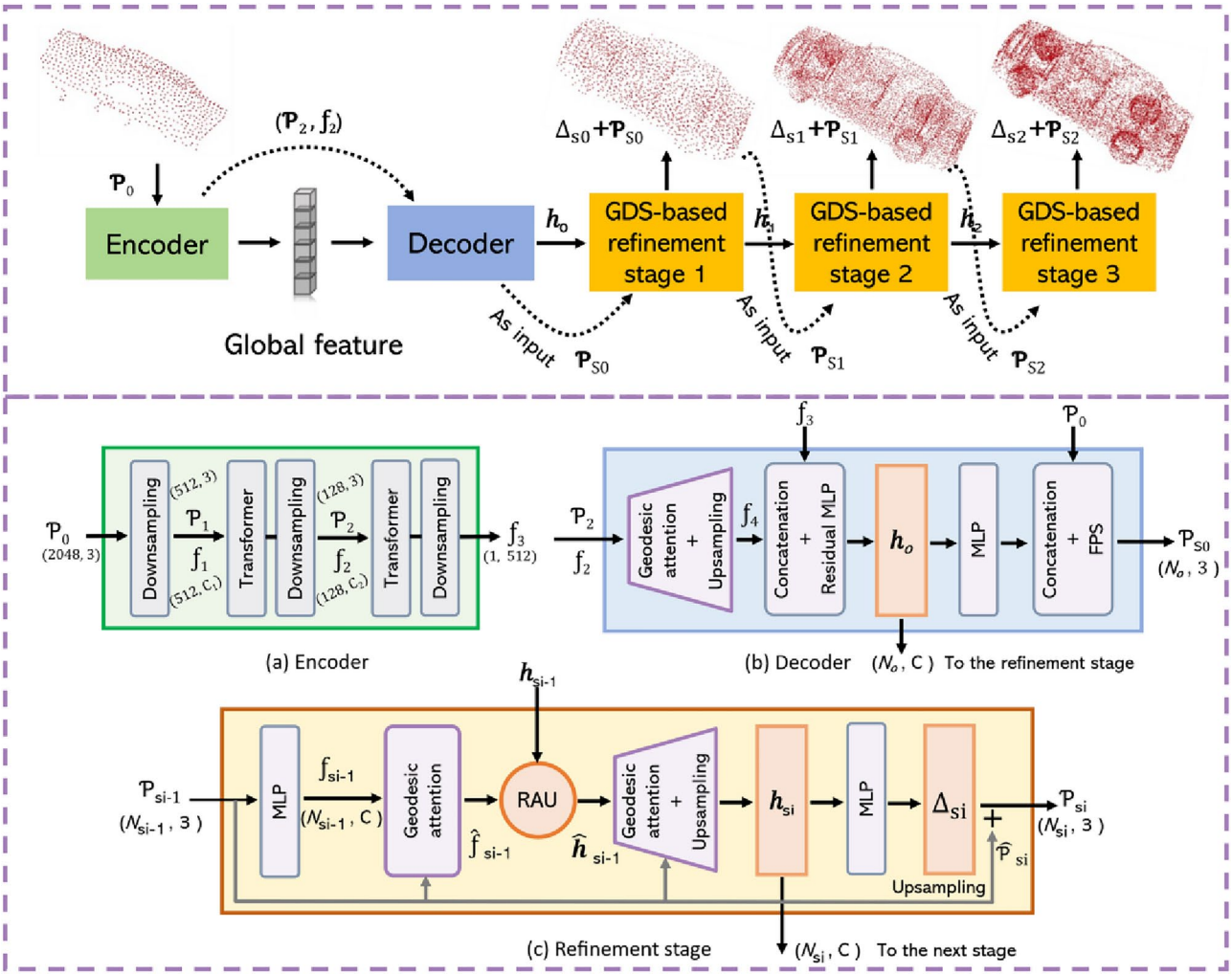


**Fig. 1**. The architecture of MRNet is shown in the upper part. The architecture mainly includes three components, an encoder for geometric feature encoding, a decoder for predicting a coarse but complete shape, and a series of refinement stages for the more detailed shape. (**a**) The encoder aims to obtain a global feature. (**b**) The decoder is used to produce a sparse point cloud. (**c**) The refinement stage aims to restore the missing details of the point cloud progressively.

cloud $\mathcal{P}_0 \in \mathbb{R}^{N\times 3}$ contains 2048 points with 3D coordinates as input, we finally obtain a global feature $\mathrm{f}_3$ as the output of the Encoder.

*Decoder*

As depicted in Fig. 1b, the Decoder takes $\mathrm{f}_2$ and $\mathcal{P}_2$ as inputs and generates coarse point features $\mathrm{f}_4$ through a geodesic distance score (GDS) based attention block with an upsampling rate $\alpha_0$. Then, the global feature $\mathrm{f}_3$ is concatenated with $\mathrm{f}_2$ and subsequently fused using a residual MLP to obtain the generated point displacement vectors (this process is repeated twice), which are used to generate a current sparse point cloud through an MLP and to initialize $h_0$. This $h_0$ serves as historical information for guiding the first refinement stage. Finally, the generated sparse point cloud is concatenated with partial point cloud $\mathcal{P}_0$ to produce $\mathcal{P}_{\mathrm{S0}}$, which represents a coarse but complete point cloud shape for the first refinement stage.

*Refinement stage*

We restore the missing details of the point cloud progressively through a series of refinement stages. Each stage receives point cloud $\mathcal{P}_{si-1}$ and point displacement vectors $h_{i-1}$ as coordinate and historical information. The output $\mathcal{P}_{si}$ is a refined point cloud. As shown in Fig. 1c, point cloud $\mathcal{P}_{si-1}$ is sent to MLP to generate point features $\mathrm{f}_{si-1}$, which are updated to $\hat{\mathrm{f}}_{si-1}$ through a GDS based attention block. Furthermore, point features $\hat{\mathrm{f}}_{si-1}$ are merged with historical information $h_{si-1}$ to $\hat{h}_{si-1}$. $\hat{h}_{si-1}$ is used to generate point displacement vectors $h_{si}$ through a GDS based attention block with an upsampling rate $\alpha_i$. Then, point displacement vectors $h_{si}$ serves as historical information for the next refinement stage and are transformed to the corresponding point displacement $\Delta_{si}$. At last, the refined point cloud is updated as $\mathcal{P}_{si} = \Delta_{si} + \hat{\mathcal{P}}_{si}$, where $\hat{\mathcal{P}}_{si}$ is the refined point cloud after interpolating $\mathcal{P}_{si-1}$.

## Geodesic attention block with upsampling

*Geodesic attention*

In the process of point cloud completion, the objective is to infer missing sections by analyzing the given geometric shape. As such, understanding the local geometric information of the object is critical for completing the missing parts. However, most current attention modules are based on Euclidean and struggle to capture local non-Euclidean geometric information accurately[32,33,46]. The geodesic represents the curve that locally minimizes distance, and the geodesic distance serves as an accurate non-Euclidean metric capable of capturing the complicated non-Euclidean geometric properties of point cloud objects. Inspired by the previous work[47], we utilize manifold-based geodesic attention to capture extended geometric relationships following the curves of implicit manifold surfaces represented by point cloud data. As shown in Fig. 2a, to compute the implicit geodesic relationships, we first apply a projection function $Proj(\cdot)$ to directly project the patch feature matrix of input point cloud $P^* = \{p_1, p_2, \cdots, p_\kappa\} \in \mathbb{R}^{K \times d}$ from Euclidean space onto an oblique manifold (OM):

$$\boldsymbol{P} := Proj(P^*) = Cat\left(\left\{\frac{p_i}{\|p_i\|}\right\}_{i=1}^{K}\right) \tag{1}$$

where $Cat(\cdot)$ corresponds to the concatenate function and $\|\cdot\|$ stands for the square Frobenius norm in the ambient space. Then, the geodesic distance between the input point pair of $\{\boldsymbol{Q}, \boldsymbol{K}\}$ on OM can be defined as follows:

$$dist(\boldsymbol{Q}, \boldsymbol{K}) = \sqrt{\sum_{i=1}^{d} arccos^2\left(diag\left(\boldsymbol{Q}^T\boldsymbol{K}\right)\right)_i} \tag{2}$$

The more related two points are, the closer their geodesic distance is. Given a point cloud's coordinates $\mathcal{P}_{\boldsymbol{xyz}}$ and its feature matrix $X$ as inputs, geodesic-based attention can be described as follows:

$$X_{out} = softmax\left(-\mathrm{GDS}(X) + \epsilon\right)(V_X + \epsilon) \tag{3}$$

$$\mathrm{GDS}(X) = dist(Q_X, K_X) \tag{4}$$

where $Q_X$, $K_X$, and $V_X$ are obtained by applying MLPs to the input, respectively, and $\epsilon$ is obtained from the position feature extractor. At the same time, we show the generation of new points under different geometric semantic scenarios in Fig. 2b.

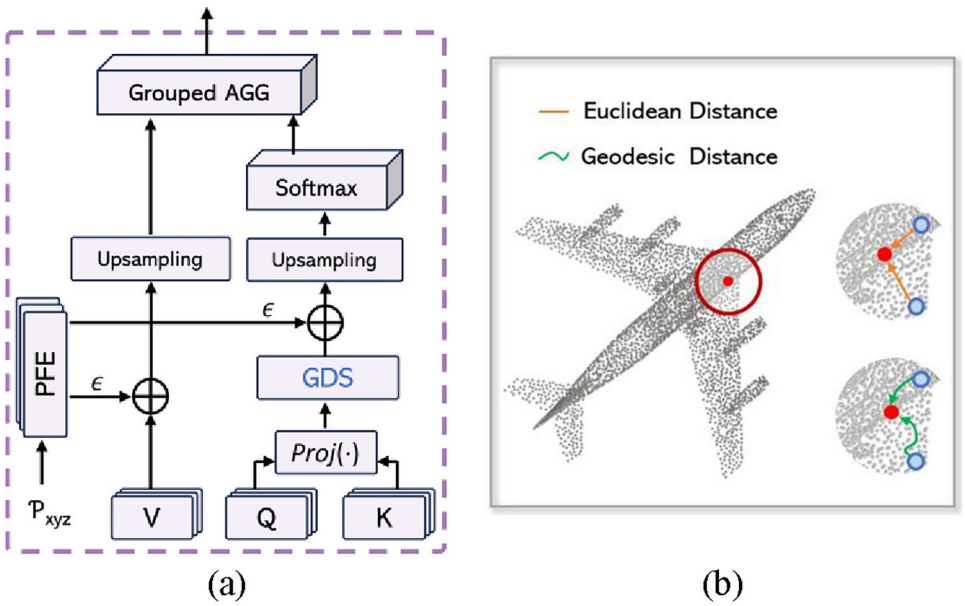


**Fig. 2**. (**a**) The pipeline of geodesic attention block with upsampling. (**b**) The generation of new points under different geometric semantic scenarios and the illustration of how neighbor point (blue) features aggregate to generate new point (red) feature.

*Upsampling*
In the $\rangle$th refinement stage, we implement interpolation and MLP to increase the number of points from $M_{i-1}$ to $M_i = \alpha_i M_{i-1}$, obtaining the values $\{v_i + \epsilon_i\}_{i=1}^{M_i}$. Then, for the center point of each point cloud patch and the values $\{v_i^\kappa + \epsilon_i^\kappa\}_{\kappa=1}^{K}$ of its $K$ neighboring points, the method attempts to implicitly learn the geodesic distance scores $\{s_i^\kappa\}_{\kappa=1}^{K}$ between the center point and its K neighbors through deconvolution before softmax. The update of a center point's feature value through the aggregation of neighboring points' feature values based on the learned geodesic distance scores can be described as follows:

$$v_i = \sum_{\kappa=1}^{K} s_i^\kappa \cdot (v_i^\kappa + \epsilon_i^\kappa) \tag{5}$$

## Position feature extractor

The distribution of points cloud is typically unordered, leading to complex geometric relationships. Previous works[32,46] based on transformer architecture have captured spatial geometric information through position encoding to mitigate feature loss in complicated transformations. As shown in Fig. 3a, the Position Feature Extractor (PFE) is further designed to update 3D coordinate information and capture non-Euclidean properties. Inspired by previous works[48], and following the assumption that a manifold locally approximates Euclidean space, we explicitly estimate the geodesic distances between all pairs of points in a patch, given the point 3D coordinates $\{p_i^\kappa\}_{\kappa=1}^{K}$ in a patch, through the following procedure on CUDA.

**Input**: 3D point coordinates $\{p_i^\kappa\}_{\kappa=1}^{K}$
**Output**: GDS Matrix
1: Construct the weighted graph G: a weighted graph connecting point $m$ and n, with weight $\|p_m - p_n\|$, if n is one of the $\sqrt{K}$ nearest point of $m$
2: Compute the matrix of shortest paths on the weighted graph as GDS matrix $\mathcal{D}_i \in \mathbb{R}^{K \times K}$
**Return** GDS matrix $\mathcal{D}_i$

**Algorithm 1**. GDS Matrix in a Patch

Finally, an MLP, denoted as $\delta$, is applied on the relative positional relationships $\Delta \boldsymbol{p}_i = \{\widehat{p}_i^\kappa\}_{\kappa=1}^{K}$, where $\widehat{p}_i^\kappa = p_i^\kappa - p_i$ and $p_i$ is the corresponding center point coordinates, to obtain the relative position features. The final position features can be described as follows:

$$\epsilon_i = \delta(\Delta \boldsymbol{p}_i) \otimes softmax(-\mathcal{D}_i) \tag{6}$$

where $\otimes$ denotes matrix multiplication.

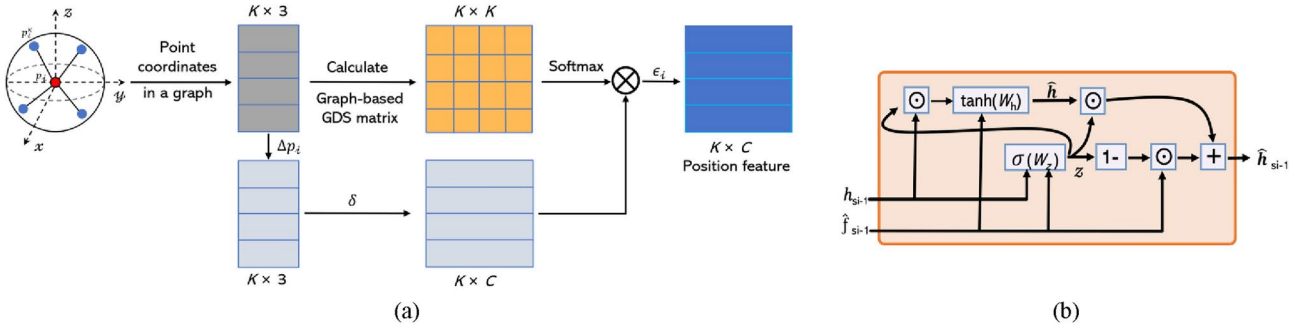


**Fig. 3**. (**a**) Details of Position Feature Extractor. $K$ represents the number of points in a patch and $C$ represents the number of dimensions in point feature space after passing a shared multi-layer perceptron $\delta$. $\otimes$ represents matrix multiplication. (**b**) Detailed illustration of data flow and operations in RAU.

### Recurrent information aggregation unit

The multi-stage refinement process can be seen as a procedure for handling sequential data, where each stage not only generates new point cloud data but also relies on the output from the previous stage. The historical information (shape point displacement features from the previous stage) provides rich geometric and semantic information about the point cloud shape, offering crucial contextual semantics for the current refinement stage.

As shown in Fig. 3b, enlightened by Gated Recurrent Unit (GRU), we proposed a Recurrent Information Aggregation Unit (RAU) for the aggregation of historical information. The RAU takes current point features $\widehat{\mathrm{f}}_{si-1}$ and historical features $h_{si-1}$ as inputs to generate the aggregated features $\widehat{h}_{si-1}$.

$$\widehat{h}_{si-1} = \mathrm{RAU}\left(\widehat{\mathrm{f}}_{si-1}, h_{si-1}\right) \tag{7}$$

Unlike the standard GRU[49], which uses two gates and emphasizes preserving previous information when calculating the output features $\widehat{h}_{si-1}$ at the current stage, the RAU uses only one update gate $z$ to balance the preservation of historical information $h_{si-1}$ with the aggregation of current point features $\widehat{\mathrm{f}}_{si-1}$ and emphasizes preserving the current point features. This process can be described as follows:

$$z = \sigma\left(W_z\left[\widehat{\mathrm{f}}_{si-1}, h_{si-1}\right] + b_z\right) \tag{8}$$

$$\widehat{h} = \tanh\left(W_h\left[z \odot h_{si-1}, \widehat{\mathrm{f}}_{si-1}\right] + b_h\right) \tag{9}$$

$$\widehat{h}_{si-1} = (1 - z) \odot \widehat{\mathrm{f}}_{si-1} + z \odot \widehat{h} \tag{10}$$

where σ is sigmoid function, and $\widehat{h}$ represents the intermediate features at current stage.

### Train loss

In training stage, we utilize Chamfer distance (CD) as the main metric to calculate completion loss[50]. We adjust the ground truth point clouds by downsampling them to correspond with the number of points at each refinement step and sum the CD loss from each stage to form the main component of the total loss. For the predicted point cloud $P$ and the corresponding ground truth $P_g$, the CD loss can be represented as follows:

$$d_{CD}\left(P, P_g\right) = \frac{1}{|P|}\sum_{x \in P}\min_{y \in P_g}\|x - y\|^2 + \frac{1}{|P_g|}\sum_{y \in P_g}\min_{x \in P}\|y - x\|^2 \tag{11}$$

Then, the total loss of the model can be defined as:

$$\mathcal{L} = \sum_{i=0}^{3}\lambda_i d_{CD}(P_i, P_g) \tag{12}$$

## Results

### Details of the implementation

The model is built using the PyTorch framework (version 2.2.1 for Linux system) from the official Pytorch website:https://pytorch.org/get-started/previous-versions/, trained on 2 NVIDIA RTX 3090 GPUs, and optimized with Adam optimizer[51] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ by totally 400 epochs. The initial learning rate is set as $5{\times}10^{-4}$ and decayed by 0.98 every 1 epoch. The batch size and weigh decay are set as 32 and $5 \times 10^{-5}$ , respectively. The intermediate model dimension $(C_1, C_2, C)$ and the number of points in a patch K are set as $(128, 256, 128)$ and 24, respectively.

### Results on the PCN dataset

Originating from the ShapeNet dataset, the widely used PCN dataset includes 30,974 models spanning eight different categories. Each complete shape contains 16,384 points uniformly sampled from the mesh surface as ground-truth and each partial shape contains 2048 points sampled from eight views as partial input. The upsampling rate for the PCN dataset is set to $(2, 1, 4, 8)$ and the L1 chamfer distance is used to evaluate the model performance. Table 1 provides the quantitative results of our approach in comparison with ten other methods on the PCN dataset. Among the ten methods, our method surpasses the top eight methods on average and all categories. Compared with PointAttN[45] and ProxyFormer[52], the lowest L1 CD in the boat category is achieved by our method.

In addition, we also visualize the completion results of different methods on six categories from the PCN dataset. We offer visual comparisons of the completion results from different methods across six categories of the PCN dataset. From Fig. 4a, we can observe that the results of PCN[26] can only recover the rough geometric structure, while other methods can recover local details well but there are noisy points. With our method, it is

| Method | Plane | Cabinet | Car | Chair | Lamp | Couch | Table | Boat | Overall |
|---|---|---|---|---|---|---|---|---|---|
| FoldingNet[34] | 9.49 | 15.80 | 12.61 | 15.55 | 16.41 | 15.97 | 13.65 | 14.99 | 14.31 |
| PCN[26] | 5.50 | 22.70 | 10.63 | 8.70 | 11.00 | 11.34 | 11.68 | 8.59 | 9.64 |
| TopNet[53] | 7.61 | 13.31 | 10.90 | 13.82 | 14.44 | 14.78 | 11.22 | 11.12 | 12.15 |
| GRNet[54] | 6.45 | 10.37 | 9.45 | 9.41 | 7.96 | 10.51 | 8.44 | 8.04 | 8.83 |
| PointTr[42] | 4.75 | 10.47 | 8.68 | 9.39 | 7.75 | 10.93 | 7.78 | 7.29 | 8.38 |
| PMP-Net++[43] | 4.39 | 9.96 | 8.53 | 8.09 | 6.06 | 9.82 | 7.17 | 6.52 | 7.56 |
| SnowflakeNet [30] | 4.29 | 9.16 | 8.08 | 7.89 | 6.07 | 9.23 | 6.55 | 6.40 | 7.21 |
| FBNet[44] | 3.99 | 9.05 | 7.90 | 7.38 | 5.82 | 8.85 | 6.35 | 6.18 | 6.94 |
| PointAttN[45] | **3.87** | **9.00** | **7.63** | 7.43 | 5.90 | **8.68** | 6.32 | 6.09 | 6.86 |
| ProxyFormer[52] | 4.01 | 9.01 | 7.88 | **7.11** | **5.35** | 8.77 | **6.03** | 5.98 | **6.77** |
| Ours | 3.98 | 9.10 | 8.19 | 7.38 | 5.41 | 8.81 | 6.30 | **5.92** | 6.89 |

**Table 1**. Point cloud completion quantitative results on the PCN dataset, measured using L1 Chamfer Distance $\times 10^3$ (lower values preferred). Significant values are in [bold].

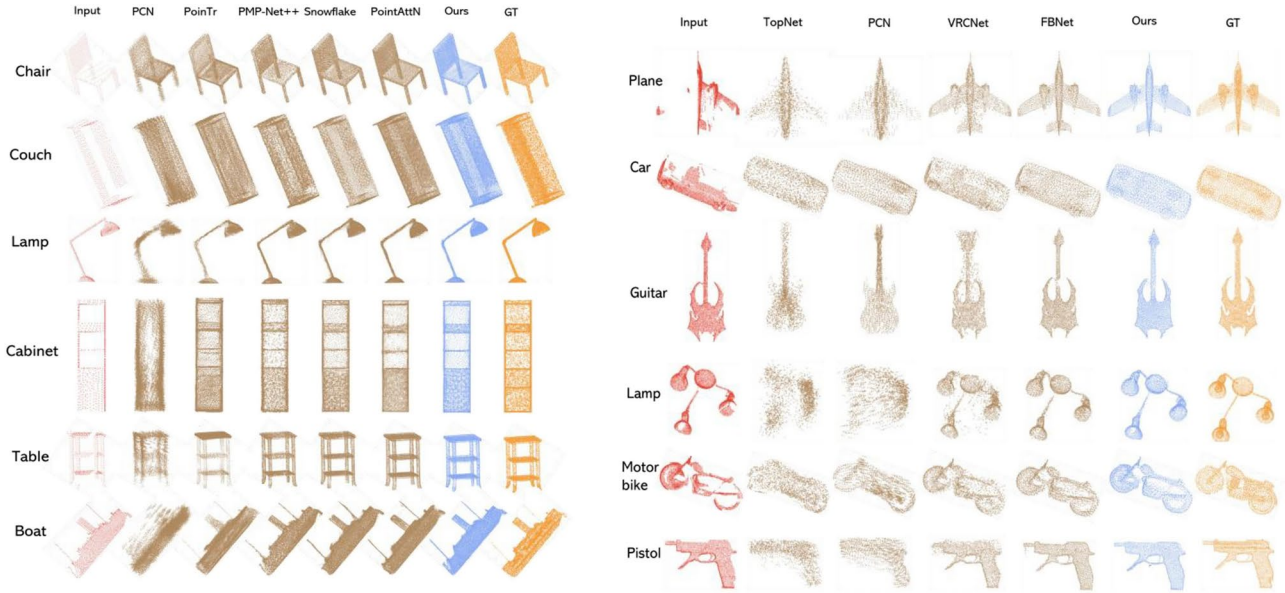


**Fig. 4**. (**a**) Qualitative assessment of point cloud completion across various methods on the PCN dataset, showing Chair, Couch, Lamp, Cabinet, Table, and Boat from top to bottom. The resolution of 16,384 points is completed. (**b**) Qualitative assessment of point cloud completion across various methods on the MVP dataset, showing Plane, Car, Guitar, Lamp, Motorbike, and Pistol from top to bottom. The resolution of 2048 points is completed.

possible to accurately infer the missing portions of the geometry of the point cloud objects, achieve finer local detail recovery, and reduce the generation of noisy points. For example, there are fewer noisy points around the global lamp shape, and the shape of the table legs and the upper part of the cabinet are close to the ground truth.

## Results on the MVP dataset

The multi-view MVP point cloud dataset includes 16 categories of point cloud shapes for training and testing. A total of 26 rendered incomplete point clouds are available for each category, obtained from uniformly spaced camera poses. The available MVP dataset contains 2048 points as complete shape. The upsampling rate for the MVP dataset is set to $(2, 1, 2, 2)$. L2 Chamfer Distance is used to quantify the geometric difference between point clouds, while F-Score assesses the balance between precision and recall in the model's output.

Table 2 provides the quantitative evaluation of our method against seven other SOTA methods on the MVP dataset. Our model stands out with the lowest L2 Chamfer Distance and the highest F1 score. Compared with the coarse-to-fine decoding methods like VRCNet[28] and SnowflakeNet[30], our MRNet achieves a reduction of L2 CD by 16.6% and 12.9%, respectively. Furthermore, in comparison to the second-best method FBNet[44], MRNet lowers the average CD error by 0.09 and boosts the F-Score by roughly 0.028, demonstrating its high efficiency.

| Method | CD | F1 |
|---|---|---|
| PCN[26] | 9.77 | 0.320 |
| TopNet[53] | 10.11 | 0.308 |
| MSN[35] | 7.90 | 0.432 |
| CRN[16] | 7.25 | 0.434 |
| VRCNet[28] | 5.96 | 0.499 |
| SnowflakeNet[30] | 5.71 | 0.503 |
| FBNet[44] | 5.06 | 0.532 |
| Ours | **4.97** | **0.538** |

**Table 2**. Quantitative results for point cloud completion on the MVP dataset in terms of L2 Chamfer Distance $\times 10^4$ (lower is better) and F1-score (higher is better). Significant values are in [bold].

| Method | Boat | Display | Jar | Stove | CD-S | CD-M | CD-H | CD-Avg | F1 |
|---|---|---|---|---|---|---|---|---|---|
| FoldingNet[34] | – | – | – | – | 26.7 | 26.6 | 40.5 | 31.2 | 0.082 |
| PCN[26] | 17.5 | 21.6 | 37.9 | 29.3 | 19.4 | 19.6 | 40.08 | 26.6 | 0.133 |
| TopNet[53] | – | – | – | – | 22.6 | 21.6 | 43.0 | 29.1 | 0.126 |
| GRNet[54] | 12.8 | 11.7 | 28.2 | 21.3 | 13.5 | 17.1 | 28.5 | 19.7 | 0.238 |
| PointTr[42] | 7.0 | 8.3 | 16.5 | 10.9 | 5.8 | 8.8 | 17.9 | 10.9 | 0.464 |
| ProxyFormer[52] | 7.1 | 7.5 | 14.8 | 10.8 | 4.9 | 7.5 | 15.5 | 9.3 | **0.483** |
| Ours | **6.3** | **7.3** | **14.6** | **10.3** | **4.6** | **6.8** | **12.6** | **8.0** | 0.461 |

**Table 3**. Point cloud completion quantitative results on the ShapeNet55 dataset, measured using L2 Chamfer Distance $\times 10^4$ (lower values preferred) and F1-score (higher values preferred). Significant values are in [bold].

| Method | 34 seen categories | | | | | 21 unseen categories | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | CD-S | CD-M | CD-H | CD-Avg | F1 | CD-S | CD-M | CD-H | CD-Avg | F1 |
| FoldingNet[34] | 18.6 | 18.1 | 33.8 | 23.5 | 0.139 | 27.6 | 27.4 | 53.6 | 36.2 | 0.095 |
| PCN[26] | 18.7 | 18.1 | 29.7 | 22.2 | 0.154 | 31.7 | 30.8 | 52.9 | 38.5 | 0.101 |
| TopNet[53] | 17.7 | 16.1 | 35.4 | 23.1 | 0.171 | 26.2 | 24.3 | 54.4 | 35.0 | 0.121 |
| GRNet[54] | 12.6 | 13.9 | 25.7 | 17.4 | 0.251 | 18.5 | 22.5 | 48.7 | 29.9 | 0.216 |
| PointTr[42] | 7.6 | 10.5 | 18.8 | 12.3 | 0.421 | 10.4 | 16.7 | 34.4 | 20.5 | 0.384 |
| ProxyFormer[52] | **4.4** | 6.7 | 13.3 | **8.1** | **0.466** | **6.0** | 11.3 | 25.4 | 14.2 | **0.415** |
| Ours | 4.9 | **6.6** | **13.2** | 8.2 | 0.457 | 6.5 | **11.1** | **23.9** | **13.8** | 0.401 |

**Table 4**. Point cloud completion quantitative results on the ShapeNet34 dataset, measured using L2 Chamfer Distance $\times 10^4$ (lower values preferred) and F1-score (higher values preferred). Significant values are in [bold].

Figure 4b shows the qualitative comparison of 2048 points completion across six categories. PCN[26] and TopNet[53] perform the worst, struggling to perceive the missing parts and even failing to maintain the original geometric structure of the input. For example, in the case of the guitar, VRCNet[28] and FBNet[44] achieve better results. However, VRCNet[28] fails to recover the neck and headstock, and FBNet[44] has a gap in the recovered neck. Obviously, our method can generate higher-quality point clouds.

## Results on the ShapeNet55/34 dataset

The ShapeNet55 dataset includes 55 categories in ShapeNet and over 50,000 objects for training and testing. The ShapeNet34 dataset includes 34 seen categories and 21 unseen categories split from original ShapeNet. Each complete shape contains 8192 points sampled from the point cloud surface. The upsampling rate for the ShapeNet55/34 dataset is set to $(2, 1, 4, 4)$. L2 Chamfer Distance is used to quantify the geometric difference between point clouds, while F-Score assesses the balance between precision and recall in the model's output.

Based on the PoinTr[42], we create different levels of incompleteness by selecting n randomly between 2048 and 6144 points in the training process. Afterward, the rest of the point clouds are downsampled to 2048 points for model input. During the testing process, eight viewpoints are fixed. Based on the value of n, set to 2048, 4096, or 6144 (25%, 50%, and 70% of the complete point cloud), the test samples are categorized into three degrees of difficulty: simple, moderate, and hard. These degrees are represented by different metrics: CD-S, CD-M and CD-H. The quantitative outcomes for our method, compared with six other methods, are displayed in Tables 3 and 4 for the ShapeNet-55 and ShapeNet-34 datasets. Table 3 lists the four categories with the best performance in the ShapeNet55 dataset. Except for the F1 score being lower than the ProxyFormer[52], our method demonstrates

| Method | FD | MMD |
|---|---|---|
| PCN[26] | 2.235 | 1.366 |
| FoldingNet[34] | 7.467 | 0.537 |
| TopNet[53] | 5.354 | 0.636 |
| GRNet[54] | 0.816 | 0.568 |
| CRN[16] | 1.023 | 0.872 |
| PointTr[42] | 0.000 | 0.526 |
| ProxyFormer[52] | 0.000 | 0.508 |
| Ours(MVP) | 0.410 | 0.640 |
| Ours(PCN) | 0.510 | 0.504 |

**Table 5**. Point cloud completion quantitative results on the KITTI dataset.

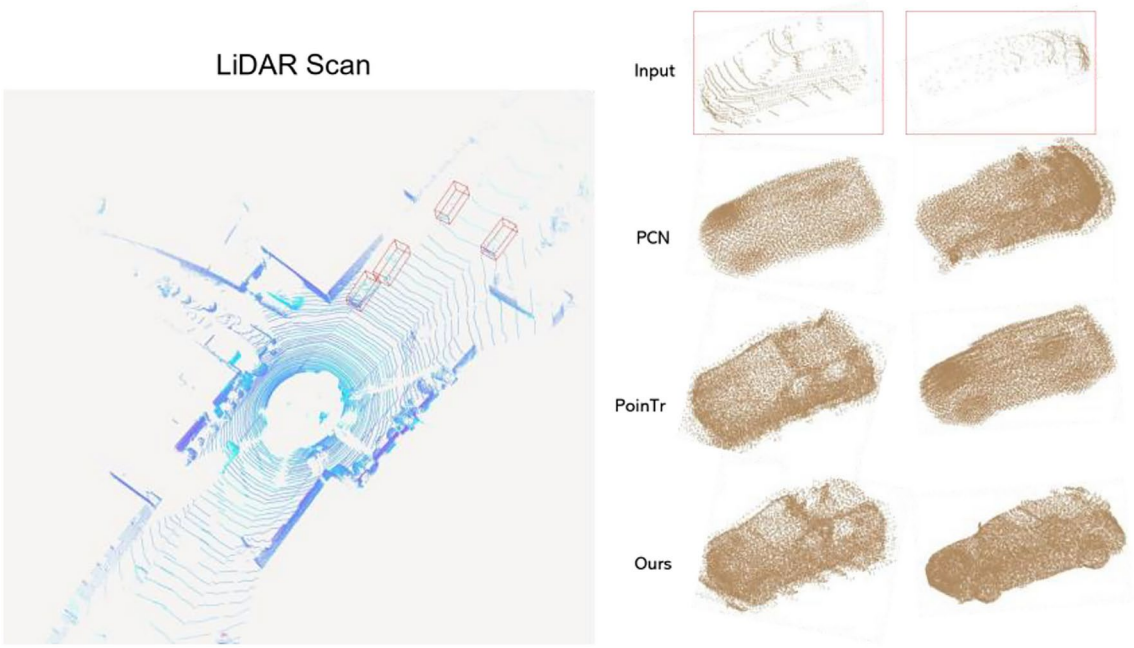


**Fig. 5**. Qualitative assessment of point cloud completion across various methods on the KITTI dataset.

superior CD results across all three difficulty levels, with the average CD being reduced by 13.9% relative to ProxyFormer[52]. Table 4 reveals that for the 34 seen categories, our method secures the lowest CD-M and CD-H by a narrow margin. Meanwhile, for the 21 unseen categories, it achieves the best results in CD-M, CD-H, and average CD, underscoring its generalization strength.

## Results on the KITTI dataset

The KITTI dataset contains 2401 real-scanned incomplete point clouds of cars by a LiDar. Since the extracted car objects of KITTI do not have ground truth shapes, the Fidelity Distance (FD) measures the accuracy of the model, while the Mini Match Distance (MMD) checks how well it matches the target data.

Following the previous work[26], we fine-tuned our pretrained model on cars in the ShapeNet dataset and MVP dataset, respectively, and evaluated it on the KITTI cars. The quantitative results, evaluated through the Fidelity Distance (FD) and Mini Match Distance (MMD) metrics for the KITTI dataset, are detailed in Table 5. PointTr[42] and ProxyFormer[52] merge input into the result, which results in an FD of 0. Then, we visualize the completion results based on the pretrained model on cars in the ShapeNet dataset, as shown in Fig. 5. Compared with PCN and PoinTr, our method can still generate a complete car point cloud on a real-world dataset, even a sparse input point cloud.

## Ablation study

To conduct a deeper analysis of the key components of MRNet for completion, we conduct a detailed ablation study on the MVP dataset.

Table 6 shows the evaluation results of different component combinations. The evaluation results of models A, B, and D show that the CD value decreases from 5.80 to 5.20, and finally to 5.01, thereby demonstrating the performance and contribution of each component. Model C and Model D compare the evaluation results of the

| Model | RAU | Dot | GDS | PFE | CD | F1 |
|---|---|---|---|---|---|---|
| A | √ | | √ | | 5.80 | 0.502 |
| B | | | √ | √ | 5.20 | 0.523 |
| C | √ | √ | | √ | 5.11 | 0.529 |
| D | √ | | √ | √ | **4.97** | **0.538** |

**Table 6**. Ablation study on MVP dataset. Significant values are in [bold].

| | Strategy | CD | F1 |
|---|---|---|---|
| w/o PFE | w/3D coordinates | 5.47 | 0.511 |
| w/PFE | Num of neighbor = 16 | 4.99 | 0.537 |
| | Num of neighbor = 20 | 5.01 | 0.538 |
| | Num of neighbor = 24 | **4.97** | **0.538** |

**Table 7**. Ablation study on position feature extractor. Significant values are in [bold].

self-attention mechanism using dot product and using geodesic distance score, respectively. Furthermore, in Table 7, we compare the Position Feature Extractor with the use of only 3D coordinates as positional information. The results demonstrate that using only 3D coordinates provides limited position feature information and setting the number of neighbors to 24 yields the best performance for our method.

## Discussion and conclusions

To further investigate the complex geometric features of point cloud objects, in this paper, we propose a geodesic attention-based multi-stage refinement transformer network named MRNet for point cloud completion. By introducing geodesic attention, we design a geodesic distance score-based attention block with upsampling to implicitly learn the geodesic distance between central and neighboring points. Meanwhile, the diagram illustrates the process of generating new points under different semantic conditions, enhancing the interpretability of the model. The weighted sum of neighboring point features is used to generate the updated central point feature. We also design a novel position feature extractor to explicitly enhance the spatial geometric perception of the non-Euclidean characteristics of point cloud objects in the point cloud completion task. Then, a Recurrent Information Aggregation Unit is further proposed to aggregate historical information from previous stages and current geometric features, guiding the feature aggregation for refinement by preserving crucial geometric structure information.

Through comprehensive experiments on PCN, MVP, ShapeNet55/34, and KITTI datasets, the proposed method has proven its effectiveness in point cloud completion when compared to state-of-the-art techniques. However, the proposed method requires a long time and high memory consumption during both the training and inference stages. The complexity of multi-stage models, especially in the point cloud completion process, is the primary reason for the defects above. In each stage of the refinement process, the model needs to restore a specific number of points in the point cloud. Additionally, the datasets used are often large, and the final point cloud contains a vast number of points. This results in an increase in the number of neurons required at each stage, which in turn raises the number of model parameters. Consequently, this generates many intermediate variables that consume significant memory, and increases both training and inference time. In future work, we may explore ways to optimize the model to reduce the time consumption and memory required for training and inference. This includes using more high-performance GPUs and distributed training, optimizing hyperparameters to reduce the memory consumption of intermediate variables, experimenting with model compression techniques, and leveraging pre-trained models to reduce training time, all while ensuring accuracy. Besides, we can extend the proposed method to other point cloud processing tasks.

## Data availability

The data are publicly available from the corresponding reference.

Received: 25 October 2024; Accepted: 12 January 2025
Published online: 28 January 2025

## References

1. Chen, Y.-N., Dai, H. & Ding, Y. Pseudo-stereo for monocular 3d object detection in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 887–897 (2022).
2. Neuville, R., Bates, J. S. & Jonard, F. Estimating forest structure from UAV-mounted LiDAR point cloud using machine learning. *Remote Sens.* **13**(3), 352. https://doi.org/10.3390/rs13030352 (2021).
3. Alexiou, E., Upenik, E. & Ebrahimi, T. Towards subjective quality assessment of point cloud imaging in augmented reality. In *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)* 1–6, (IEEE, 2017).
4. Tran, A. T., Le, H. S., Lee, S. H. & Kwon, K. R. Pointct: Point central transformer network for weakly-supervised point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3556–3565 (2024)

5. Yew, Z. J. & Lee, G. H. Rpm-net: Robust point matching using learned features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11824–11833 (2020).
6. Shi, W. & Rajkumar, R. Point-gnn: Graph neural network for 3d object detection in a point cloud. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 1711–1719. (2020).
7. Berger, M. et al. A survey of surface reconstruction from point clouds. *Comput. Graph. Forum* **36**(1), 301–329 (2017).
8. Brock, A., Lim, T., Ritchie, J. M., & Weston, N. Generative and discriminative voxel modeling with convolutional neural networks. arXiv:1608.04236 (2016).
9. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., & Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1912–1920 (2015).
10. Le, T., & Duan, Y. Pointgrid: A deep network for 3d shape understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9204–9214. (2018).
11. Dai, A., Ruizhongtai Qi, C. & Nießner, M. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5868–5877. (2017).
12. Sinha, A., Unmesh, A., Huang, Q., & Ramani, K. Surfnet: Generating 3d shape surfaces using deep residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* 6040–6049 (2017).
13. 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*, 652–660 (2017).
14. Gu, J., Ma, W.C., Manivasagam, S., Zeng, W., Wang, Z., Xiong, Y., Su, H. & Urtasun, R. "Weakly-supervised 3d shape completion in the wild. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, 283–299 (Springer, 2020).
15. Huang, Z., Yu, Y., Xu, J., Ni, F., & Le, X. Pf-net: Point fractal network for 3d point cloud completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7662–7670 (2020).
16. Wang, X., Ang Jr, M. H. & Lee, G. H. *Cascaded Refinement Network for Point Cloud Completion*. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 790–799 (2020).
17. Qi, C. R., Yi, L., Su, H. & Guibas, L. J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, vol 30 (2017).
18. 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*, 2790–2799. (2018).
19. Wang, Y. et al. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph. (tog)* **38**(5), 1–12 (2019).
20. Wang, K., Chen, Ke. & Jia, K. Deep cascade generation on point sets. *In IJCAI* **2**, 4 (2019).
21. Shi, J., Lingyun, Xu., Heng, L. & Shen, S. Graph-guided deformation for point cloud completion. *IEEE Robot. Automat. Lett.* **6**(4), 7081–7088 (2021).
22. Zhu, L., Wang, B., Tian, G., Wang, W. & Li, C. Towards point cloud completion: Point rank sampling and cross-cascade graph cnn. *Neurocomputing* **461**, 1–16 (2021).
23. Miao, Y., Zhang, L., Liu, J., Wang, J. & Liu, F. An end-to-end shape-preserving point completion network. *IEEE Comput. Graphics Appl.* **41**(3), 20–33 (2021).
24. Mendoza, A., Apaza, A., Sipiran, I. & Lopez, C. Refinement of predicted missing parts enhance point cloud completion. arXiv:2010.04278 (2020).
25. Sarmad, M., Lee, H. J. & Kim, Y. M. Rl-gan-net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5898–5907 (2019).
26. Yuan, W., Khot, T., Held, D., Mertz, C., & Hebert, M. Pcn: Point completion network. *In 2018 International Conference on 3D Vision (3DV)*, pp. 728–737, 2018.
27. Wen, X., Xiang, P., Han, Z., Cao, Y. P., Wan, P., Zheng, W. & Liu, Y. S. Pmp-net: Point cloud completion by learning multi-step point moving paths. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7443–7452 (2021).
28. Pan, L., Chen, X., Cai, Z., Zhang, J., Zhao, H., Yi, S. & Liu, Z. Variational relational point completion network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8524–8533 (2021).
29. Zhang, X., Feng, Y., Li, S., Zou, C., Wan, H., Zhao, H., Guo, Y. & Gao, Y. View-guided point cloud completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15890–15899 (2021).
30. Xiang, P., Wen, X., Liu, Y. S., Cao, Y. P., Wan, P., Zheng, W., & Han, Z. Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5499–5509 (2021).
31. Huang, T., Zou, H., Cui, J., Yang, X., Wang, M., Zhao, X., Zhang, J., Yuan, Y., Xu, Y. and Liu, Y. RFNet: Recurrent forward network for dense point cloud completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 12508–12517 (2021).
32. Wei, X., Wang, M., Lin, S. H. J., Li, Z., Yang, J., Al-Jawari, A. & Tang, X. Multi-scale Geometry-aware Transformer for 3D Point Cloud Classification. arXiv:2304.05694 (2023).
33. He, T., Huang, H., Yi, L., Zhou, Y., Wu, C., Wang, J. & Soatto, S. Geonet: Deep geodesic networks for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6888–6897. (2019).
34. 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*, 206–215 (2018).
35. Liu, M., Sheng, Lu., Yang, S., Shao, J. & Shi-Min, Hu. Morphing and sampling network for dense point cloud completion. *Proc. AAAI Conf. Artif. Intell.* **34**(07), 11596–11603 (2020).
36. Wen, X., Li, T., Han, Z. & Liu, Y. S. Point cloud completion by skip-attention network with hierarchical folding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1939–1948 (2020).
37. Nie, Y. et al. Skeleton-bridged point completion: From global inference to local adjustment. *Adv. Neural. Inf. Process. Syst.* **33**, 16119–16130 (2020).
38. Xia, Y., Xia, Y., Li, W., Song, R., Cao, K. and Stilla, U. Asfm-net: Asymmetrical siamese feature matching network for point completion. In *Proceedings of the 29th ACM international conference on multimedia*, 1938–1947 (2021).
39. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. & Polosukhin, I.Attention is all you need. *Advances in neural information processing systems*, vol 30 (2017).
40. Guo, M.-H. et al. Pct: Point cloud transformer. *Comput. Vis. Media* **7**, 187–199 (2021).
41. Zhao, H., Jiang, L., Jia, J., Torr, P. H. S. & Koltun, V. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16259–16268 (2021).
42. Yu, X., Rao, Y., Wang, Z., Liu, Z., Lu, J. & Zhou, J. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, 12498–12507 (2021).
43. Wen, X. et al. Pmp-net++: Point cloud completion by transformer-enhanced multi-step point moving paths. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(1), 852–867 (2022).
44. Yan, X., Yan, H., Wang, J., Du, H., Wu, Z., Xie, D., Pu, S. & Lu, L. Fbnet: Feedback network for point cloud completion. In *European Conference on Computer Vision*, 676–693 (Springer, 2022).
45. Wang, J. et al. Pointattn: You only need attention for point cloud completion. *Proc. AAAI Conf. Artif. Intell.* **38**(6), 5472–5480 (2024).
46. Li, Z. et al. Geodesic self-attention for 3d point clouds. *Adv. Neural. Inf. Process. Syst.* **35**, 6190–6203 (2022).

47. Qi, G., Yu, H., Lu, Z. & Li, S. Transductive few-shot classification on the oblique manifold. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* 8412–8422 (2021).
48. Whiteley, N., Gray, A. & Rubin-Delanchy, P. Matrix factorisation and the interpretation of geodesic distance. *Adv. Neural. Inf. Process. Syst.* **34**, 24–38 (2021).
49. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv:1406.1078. (2014).
50. 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*, 605–613. (2017).
51. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. arXiv:1412.6980. (2014).
52. Li, S., Gao, P., Tan, X. & Wei, M. Proxyformer: Proxy alignment assisted point cloud completion with missing part sensitive transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9466–9475 (2023).
53. Tchapmi, L. P., Kosaraju, V., Rezatofighi, H., Reid, I., & Savarese, S. Topnet: Structural point cloud decoder. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 383–392 (2019).
54. Xie, H., Yao, H., Zhou, S., Mao, J., Zhang, S. & Sun, W. Grnet: Gridding residual network for dense point cloud completion. In *European Conference on Computer Vision*, 365–381 (Springer, 2020).

## Acknowledgements

This research was funded by grants from the National Natural Science Foundation of China: Special Program in AI (52441205), Shaanxi Provincial Department of Science and Technology Key R&D Project-Key Core Technology Research Project (2024SF2-GJHX-46), National Key R&D Program of China (2022YFC3801300) and a cooperative project with CCTEG China Coal Mining Research Institute (TDKC-DZ-2022-016).

## Author contributions

Yuchen Chang: Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Writing-original draft, Writing-review & editing. Kaiping Wang: Conceptualization, Funding acquisition, Supervision, Writing-review & editing.

## Declarations

## Competing interests

The authors declare no conflicts of interest.

## Additional information

**Correspondence** and requests for materials should be addressed to K.W.