





TGPS: dynamic point cloud down-sampling of the dense point clouds for Terracotta Warrior fragments

JIE LIU,^{1,2,†} DA SONG,^{1,2,†} GUOHUA GENG,^{1,2,3}  YU TIAN,^{1,2}
MENGNA YANG,^{1,2} YANGYANG LIU,^{1,2} MINGQUAN ZHOU,^{1,2} KANG
LI,^{1,2} AND XIN CAO^{1,2,4} 

¹School of Information Science and Technology, Northwest University, Xi'an, Shaanxi, China

²National and Local Joint Engineering Research Center for Cultural Heritage Digitization, Xi'an, Shaanxi 710127, China

³ghgeng@nwu.edu.cn

⁴xin_cao@163.com

[†]Authors contributed equally to this work.

Abstract: The dense point clouds of Terracotta Warriors obtained by a 3D scanner have a lot of redundant data, which reduces the efficiency of the transmission and subsequent processing. Aiming at the problems that points generated by sampling methods cannot be learned through the network and are irrelevant to downstream tasks, an end-to-end specific task-driven and learnable down-sampling method named TGPS is proposed. First, the point-based Transformer unit is used to embed the features and the mapping function is used to extract the input point features to dynamically describe the global features. Then, the inner product of the global feature and each point feature is used to estimate the contribution of each point to the global feature. The contribution values are sorted by descending for different tasks, and the point features with high similarity to the global features are retained. To further learn rich local representation, combined with the graph convolution operation, the Dynamic Graph Attention Edge Convolution (DGA EConv) is proposed as a neighborhood graph for local feature aggregation. Finally, the networks for the downstream tasks of point cloud classification and reconstruction are presented. Experiments show that the method realizes the down-sampling under the guidance of the global features. The proposed TGPS-DGA-Net for point cloud classification has achieved the best accuracy on both the real-world Terracotta Warrior fragments and the public datasets.

© 2023 Optica Publishing Group under the terms of the [Optica Open Access Publishing Agreement](#)

1. Introduction

As a critical carrier of Chinese history and culture, the virtual restoration of the Terracotta Warriors has important social and economic value for the digital protection of cultural relics and the study of cultural inheritance. With the emergence of 3D laser scanner technology, point clouds are easily obtained and widely used in computer vision (CV), such as automatic driving [1], remote sensing [2], cultural heritage protection [3,4,5,6,7] and other fields. The digital collection of the Terracotta Warrior fragments can be achieved by a 3D scanner and restored virtually. The dense point clouds have lots of redundant points and need excessively large storage space and a lot of time for post-processing. It is not conducive to the storage, transmission, calculation, and display of cultural relics. Therefore, how to accurately and effectively simplify the dense point clouds while maintaining the feature points of models is still a challenging problem for virtual restoration of the cultural relics artifacts.

Due to the sparseness, unstructuredness, and irregularity of point clouds, the traditional Convolutional Neural Network (CNN) is not suitable for the point clouds directly. As a pioneering work, PointNet [8] can perform analysis directly on point cloud using deep neural

networks. To learn richer local structures, many follow-up works are improved which can generally be divided into neighborhood feature fusion [9,10], graph-based convolution [11,12,13], and kernel-based convolution [14,15]. PointNet++ [9] builds a hierarchical feature learning structure. To collect critical points from the upper layer, the intervention of down-sampling methods is required. PointNet++ first utilizes Farthest Point Sampling (FPS) to select centroids at different levels, then query the neighbor points of the centroids within a specific radius to form local regions. However, the FPS is only based on the Euclidean distance with high time complexity and not suitable for the sparsity and semantic information of point clouds. Subsequently, some researchers have attempted to design apply deep learning to the study of the point cloud down-sampling [16,17,18,19,20]. However, The existing approaches are not suitable for the large and dense point clouds of the Terracotta Warrior fragments, which have tens of thousands of points. At the same time, these task-independent sampling network performance still need to further improve.

To address the problems mentioned above, an end-to-end learnable down-sampling method named TGPS for the Terracotta Warrior fragments is proposed. The overall framework of the method is illustrated in Figure 1. First, we divide the dense point clouds into irregular local patches (may overlap) using FPS and K-Nearest Neighborhood (KNN) algorithm. Second, the TGPS can produce sampled points which related to the different downstream tasks. Moreover, the DGA EConv aggregates the features of a set of points' contribution coefficients to further enrich the local region information. Third, the TGPS-DGA Module, which is consisted of TGPS and DGA EConv, is followed by a folding-based decoder [11] (or a fully connected (FC) layer) forming a completed network for point cloud reconstruction (or classification). Finally, our equipped model is evaluated on the public datasets and the real-world datasets. Experiments demonstrate that the proposed method achieves comparable performance with other approaches in point cloud reconstruction and classification. The main contributions are summarized as follows:

- We propose an end-to-end specific task-driven and learnable down-sampling approach (TGPS) based on Transformers, and present the sampling results for the downstream tasks for 3D Terracotta Warrior reconstruction and classification. Experiments show our method dose well for the large and dense point clouds of the Terracotta Warrior fragments.
- We propose a DGA EConv for local feature aggregation which exploits the attention mechanism on a local graph to capture accurate and robust geometric details of Terracotta Warrior fragments.

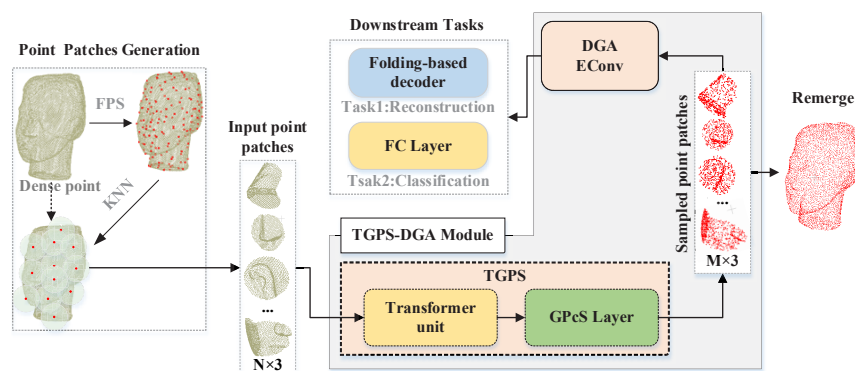


Fig. 1. Overview of the proposed method.

- We propose a TGPS-DGA-Net, which is equipped with the TGPS-DGA Module and the FC layer, for 3D object classification that achieves the best accuracy on the real-world Terracotta Warrior fragments and the public datasets.

2. Related work

2.1. Point cloud sampling

Since FPS [9] and SO-Net [10] both are offline and task-independent down-sampling methods. Subsequently, some methods have been improved to a certain extent. Yang et al. [16] utilized a Self-Attention (SA) mechanism to learn the relationship between points and used Gumbel Subset Sampling (GSS) to improve the accuracy. S-Net [17] generated the sampled points optimized for a specific task. However, it was not differentiable in the matching operation and could not propagate gradients through the neural network. Furthermore, Lang et al. [18] proposed a differentiable method to approximate point cloud sampling. To address the problem that existing down-sampling methods did not consider the importance of the sampled points to the final output, Nezhadarya et al. [19] presented an adaptive down-sampling layer named Critical Points Layer (CPL), which could simplify the unordered point clouds while preserving critical points. However, the sampling task was trained separately and the performance was limited. Wang et al. [20] proposed a trainable method (GDS) for the point cloud classification task without repetitive sampling and introduced the Chamfer distance (CD) loss function to make the sampling results as uniform as possible. Our method can be regarded as an improvement to the GDS in some extent, and the sampling results are more important for downstream tasks.

2.2. Transformers

Transformers were first proposed and had achieved great success in the field of Natural Language Processing (NLP) [21,22]. Then, they have also shown promising performance on various vision tasks [23,24,25,26,27]. More recently, there have also been attempts to apply Transformers to point cloud data. PT [28] and PCT [29] achieved the superior performance of visual Transformers in point cloud analysis. Subsequently, some scholars extended Transformer to various applications, such as 3DETR [30], Pointformer [31] for 3D object detection, PTTR [32] for 3D target tracking, and PST-NET [33] for point cloud simplification, etc. Yet a learned point clouds down-sampling method based on Transformers, subject to a subsequent task objective, has not been proposed before. Our TGPS can be seen as a new attempt approach.

2.3. Graph convolution methods

Graph convolutional networks can be categorized as spectral approaches [34,35,36] and non-spectral approaches [12,37,38]. As the Eigen decomposition of the Laplacian operator requires high computing costs, and the spectral approaches require a large number of parameters and difficult spatial positioning. Different from spectral approaches, the non-spectral method describes the spatial relationship of each point with vertices and edges, and constructs a convolution kernel by aggregating edge features and vertex features. Simonovsky et al. [37] proposed the edge convolutional operations performs on graph signals in the spatial domain. However, the edge label generation process is dynamic, the irregular local point distribution is not considered. DGCNN [12] adopted EdgeConv to capture the local relationship and dynamically updated the relationship graph. LDGCNN [38] increased the number of network layers and uses residual connections to improve the model performance, but directly using the max-pooling would lead to the loss of some important information. Therefore, we propose the DGA EConv that can learn local relations between points.

3. Methods

Given a dense point cloud $D \in G \times 3$, a set of local patches $\{P_i\} \subseteq N \times 3$, $i = 1, 2, \dots, C$, are obtained by FPS and KNN, where C is the number of local patches. To alleviate the overfitting of the network, each local patch with N is first subjected to perform conventional data augmentation (e.g. rotation or jittering) to improve the diversity of the training samples, while preserving the structure of the raw data. Next, the features are encoded by the Transformer unit, and the sampled points can be obtained by the TGPS. The number of sampled points of each patch is M . Then, the DGA EConv is used to aggregate the center point features by the importance of the neighbor points to enrich the local region. Finally, combined the TGPS-DGA Module with FC layers to form TGPS-DGA-Net for 3D object classification. The overall framework of TGPS-DGA-Net is illustrated in Figure 2. We evaluate our pretrained models on the real-world Terracotta Warrior fragments and the public datasets, e.g., ModelNet and ScanObjectNN, and achieve the best accuracy among the comparative methods.

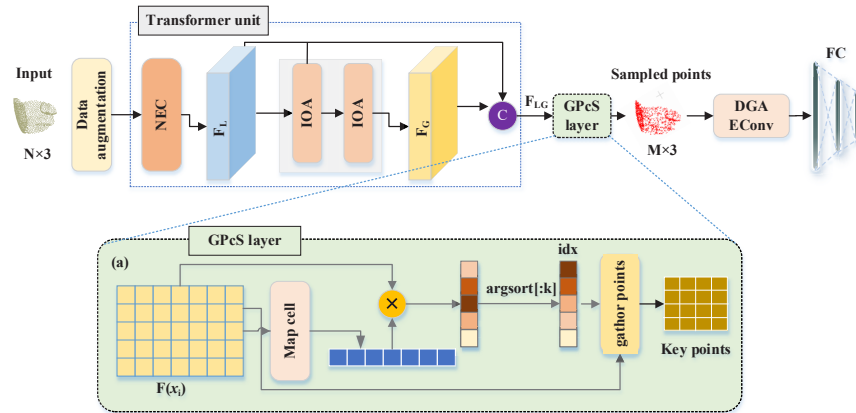


Fig. 2. Illustration of the TGPS-DGA-Net.

3.1. Transformer unit

We follow the coordinate-based input embedding method in [34]. The input points are first embedded into a high-dimensional space through MLP, and then the Neighbor embedding cell (NEC) is used to extract local features. Next, the global features are obtained by two consecutive Improved Offset-Attention (IOA) cells to obtain the global features to learn the rich semantic features of each point. The output feature vector of Transformer unit is obtained by the concatenation of local features and global features. Hence, the Transformer unit can learn more about geometric information and semantic information, and improve the ability to perceive the local features.

3.1.1. Neighbor embedding cell

Neighbor embedding cell is used to strengthen the capability of local feature extraction. After data augmentation, the input point cloud P with N points and corresponding features F , and the sampled point clouds $P_S \in P$ are obtained by down-sampling. For each point p_i of the sampled point cloud, the neighborhood $N(p_S) = \{p_S^j | j = 1, 2, \dots, k\}$ is gathered by the KNN. Then, the difference feature between the center point p_S and its neighbor point p_S^j are calculated as the local feature, and the MLP is combined with the point cloud feature before neighborhood aggregation. Finally, the aggregated feature $F_{NS}(p_S)$ of the neighbor embedding cell are generated after the

max-pooling.

$$F_{NS}(p_S) = M(MLP(C(C(F(p_S^j) - F(p_S)), R(F(p_S), k)))) \quad (1)$$

where $F(p_S)$ is the corresponding output feature of the point p_S , $C(\cdot)$ denotes concatenation operation, $R(F(p_S), k)$ is the operation that repeats the input $F(p_S)$ k times to construct a matrix, $M(\cdot)$ is the max-pooling operation.

3.1.2. Improved offset-attention (IOA)

As a core component of Transformers, the SA is used to establish the inner links between features so that the similar features can be selectively aggregated together. In our TGPS-DGA Module, an improved Offset-Attention (IOA) is introduced to replace the original SA, which can improve the performance of the network. First, a set of query, key, and value pairs is produced by linear transformations of the input feature $F_{NS}(p_S)$.

$$(Q, K, V) = F_{NS}(p_S) \cdot (W_q, W_k, W_v) \quad (2)$$

where $Q, K \in \mathbb{R}^{N_S \times d_u}$, $V \in \mathbb{R}^{N_S \times d_i}$, $W_q, W_k \in \mathbb{R}^{d_i \times d_u}$, $W_v \in \mathbb{R}^{d_i \times d_i}$, W_q, W_k, W_v denotes the learnable linear transformation weights, d_u is the dimension of the matrices Q and K , d_i is the dimension of the matrix V . For less computation, the dimension d_i is set to be $d_u/4$. We compute the dot products of the query points with key points, with scaling by $\sqrt{d_u}$, and apply a softmax function to obtain the weights on the values. The higher the attention weight, the stronger the relevance of the feature. Next, the weighted sum of the attention weights and the corresponding values in matrix V is calculated and the final output features are denoted by $F_{sa}(p_S)$.

$$F_{sa}(p_S) = SA(Q, K, V) = \text{soft max}\left(\frac{QK^T}{\sqrt{d_u}}\right)V \quad (3)$$

The IOA cell can be built upon the different relationship functions between the SA features $F_{sa}(p_S)$ and the input features $F_{NS}(p_S)$ for each point. To avoid the disappearance of the gradient during training, the features are passed through the MLP and the final output features $F_{ioa}(p_S)$ of the IOA are obtained by skip connections with the feature $F_{NS}(p_S)$.

$$F_{ioa}(p_S) = IOA(F_{NS}(p_S)) = MLP(\theta(F_{NS}(p_S), F_{sa}(p_S))) + F_{NS}(p_S) \quad (4)$$

where $F_{ioa}(p_S)$ denotes the output feature of a single-layer IOA, $\theta(\cdot)$ is a relational function.

As shown in Figure 3, the number 1 to 5 represents each of the five different relational functions, which is Summation, Subtraction, Concatenation, Hadamard product, and Dot product, respectively.

- (a) Summation: $\theta(p_i, p_{ij}) = o(f(p_i)) + \mu(f(p_{ij}))$
- (b) Subtraction: $\theta(p_i, p_{ij}) = o(f(p_i)) - \mu(f(p_{ij}))$
- (c) Concatenation: $\theta(p_i, p_{ij}) = [o(f(p_i)), \mu(f(p_{ij}))]$
- (d) Hadamard product: $\theta(p_i, p_{ij}) = o(f(p_i)) \odot \mu(f(p_{ij}))$
- (e) Dot product: $\theta(p_i, p_{ij}) = o(f(p_i)) \cdot \mu(f(p_{ij}))$

where $o(\cdot)$ and $\mu(\cdot)$ are trainable transformations, such as the form of linear functions. As shown in Table 6, the relation function $\theta(\cdot)$ takes the best result of the Concatenation. Please see more discussion and analysis in Section 4.4.2 for details.

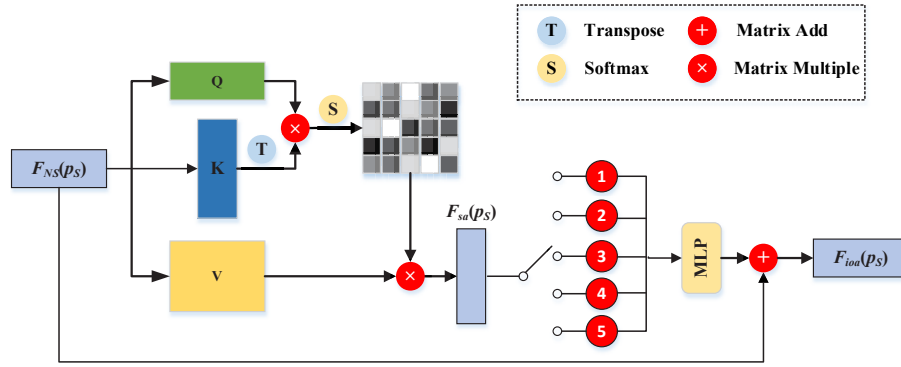


Fig. 3. Improved Offset Attention Cell.

Finally, the output feature $F_G(p_S)$ of two consecutive IOA cells is expressed as Eq. (7).

$$F_{ioa}^1(p_S) = IOA^1(F_{NS}(p_S)) \quad (5)$$

$$F_{ioa}^2(p_S) = IOA^2(F_{ioa}^1(p_S)) \quad (6)$$

$$F_G(p_S) = C(F_{ioa}^1(p_S), F_{ioa}^2(p_S)) \cdot W_o \quad (7)$$

where IOA^i denotes the i th improved offset attention layer, each attention layer has the same output dimension as its input and W_o is the weight of the linear layer.

The output feature $F_{LG}(p_S)$ of Transformer unit is formed by concatenating the local feature $F_L(p_S)$ and global feature $F_G(p_S)$.

3.2. GPcS layer

As shown in the GPcS layer in Figure 2(a), the output feature of the Transformer unit $F_G(p_S)$ is fed into a mapping function ϕ , which is consisted of the MLP followed by a global max-pooling, to learn a score vector $F_\mu(p_S)$ to estimate each point's significance $\gamma(p_S)$. The score vector $F_\mu(p_S)$ can be seen as an optimizable variable related to the description of the input cloud during training. Therefore, the score vector $F_\mu(p_S)$ can be expressed as:

$$F_\mu(p_S) = \phi(F_G(p_S)) = M(MLP(F_G(p_S))) \quad (8)$$

The significance of each point $\gamma(p_S)$ is calculated by the inner product between $F_G(p_S)$ and $F_\mu(p_S)$, which can be regarded as the semantic similarity between each point and global feature as follows:

$$\gamma(p_S) = F_G(p_S) \times F_\mu(p_S)^T \quad (9)$$

The features of important points and their coordinates are preserved for global shape description by retaining points with high scores. As shown in Eq. (10), we gather the index of top- m ranking points in the value of $\gamma(p_S)$ and the corresponding features form F_g . The characteristics of the sampled point can be obtained by Eq. (11).

$$idx = \arg \text{sort}(\gamma(p_S))[: M] \quad (10)$$

$$F_g(p_S) = F_G(idx, :) \quad (11)$$

As the sampling rate gradually increases, this will result in the loss of a large amount of local geometric detail. Therefore, the spatial neighborhood features of the downsampled points need to be aggregated. To enrich the feature information, the features of the previous layer of the

sampled points are included. The simplest solution is to find the input feature $F_G(p_S)$ of the corresponding sampled point from the input point clouds. The sampled point feature $F_g(p_S)$ and the local geometric details aggregated by the KNN algorithm are connected to obtain rich local features F_n , as shown in Eq. (12).

$$F_n(p_S)[i] = \left(\max_{j \in \text{Nei}(i)} F_G(p_S)[j] \right) \oplus F_g(p_S)[i] \quad (12)$$

3.3. Dynamic graph attention edge convolution

For a given graph $G = (V, E)$, where the set of vertices is $V = \{p_i, p_{j1}, p_{j2}, \dots, p_{jk}\}$ and the set of edges is $E \subseteq V \times V$. For computation efficiency, we use the KNN algorithm to construct a neighborhood graph for the center point p_i and the neighborhood point p_{ij} . The edge set E between nodes is composed of a set of directed edges $\{(p_i, p_{i1}), (p_i, p_{i2}), \dots, (p_i, p_{ik})\}$. To make the center point p_i pay more attention to the points with larger contributions, it is necessary to aggregate the point features according to the importance of the neighbor points. Inspired by [12,37,39], we propose a DGA EConv which takes into account the local geometric structure of the points. The DGA EConv extracts features by three steps: (1) Construct a graph using the KNN algorithm. (2) For each point, calculate the attention score of its neighbor points. (3) Calculate the weighted average of the feature information of the neighbor points according to the obtained attention score. As a core component of the proposed framework, the DGA EConv can perform 3D point cloud classification and reconstruction based on local attention.

The main purpose of dynamic graph attention convolution is to learn a function $g : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$, where d and d' are the feature dimensions. The input feature $H = \{h_1, h_2, \dots, h_N\}$, $h_i \in \mathbb{R}^d$ is embedded into a higher dimensional feature $H' = \{h'_1, h'_2, \dots, h'_N\}$ through function $g(\cdot)$, where $h'_i \in \mathbb{R}^{d'}$. By reasonably assigning weights to the edge features formed by the k-nearest neighbor graph, the interference of distant points is weakened, and the features of close points are relatively strengthened, which helps the network to better learn suitable local features. To reasonably assign weights to the point clouds, a shared attention convolution kernel $r : \mathbb{R}^{3+d} \rightarrow \mathbb{R}^{d'}$ is constructed, which is obtained by learning the features of the center point p_i and its neighbor point p_{ij} . The attention edge coefficient is defined as w'_{ij} , which represents the importance of the neighbor point to the center point. The coefficient w'_{ij} is calculated by the attention mechanism r .

$$w'_{ij} = r(C((p_{ij} - p_i), (\omega_g(h_{ij}) - \omega_g(h_i)))) \quad (13)$$

where $w'_{ij} = [w'_{ij,1}, w'_{ij,2}, \dots, w'_{ij,k}] \in \mathbb{R}^{d'}$, $\omega_g(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$ is a feature mapping function, an MLP maps input point features to high-dimensional features. $(p_{ij} - p_i)$ represents the relative spatial relationship between point p_i and its neighbor point p_{ij} , h_{ij} represents the input feature of the neighborhood point p_{ij} , $(\omega_g(h_{ij}) - \omega_g(h_i))$ represents the feature difference between two points, and $C(\cdot)$ denotes concatenation operation. From Eq. 1, the attention edge coefficient will assign more attention to similar neighborhood points, which are similar in both distance and feature space. To make the edge coefficients easy to compare with neighbors of different scale sizes at different points, they are normalized using the softmax function.

$$w_{ij} = \text{soft max}(w'_{ij}) = \frac{\exp(w'_{ij,k})}{\sum_{j \in N(i)} \exp(w'_{ij,k})} \quad (14)$$

where $w'_{ij,k}$ represents the attention weight of p_{ij} corresponding to p_i on the kth feature channel. The shared attention convolution r can be implemented by a MLP. Expanding Eq. 14, the final

normalized edge coefficients can be expressed as:

$$w_{ij} = \frac{\exp(\omega_\alpha(C((p_{ij,k} - p_{i,k}), \omega_g(h_{j,k}) - \omega_g(h_{i,k}))))}{\sum_{j \in N(p_i)} \exp(\omega_\alpha(C((p_{ij,k} - p_{i,k}), \omega_g(h_{j,k}) - \omega_g(h_{i,k}))))} \quad (15)$$

where ω_α stands for the MLP. The normalized edge coefficient w_{ij} is used to assign weights to each edge, and the weighted sum of the neighborhood point features of the central point p_i is calculated as the final output feature for each point. The DGA EConv calculation operator is shown in Figure 4. The final updated features of point p_i can be defined as:

$$h_i^{DGA} = \sum_{j \in N(i)} w_{ij} \omega_g(h_j) + b_i \quad (16)$$

where $\omega_g(h_j)$ is the feature mapping function and b_i is the corresponding bias term.

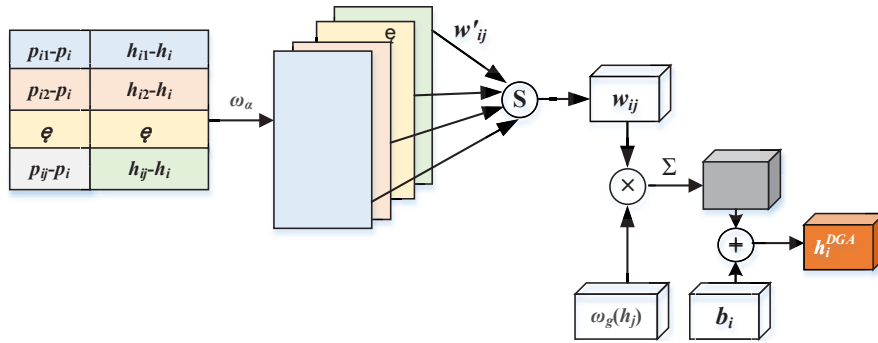


Fig. 4. Illustration of DGA EConv operator.

3.4. Loss

CD loss L_{cd} is to force the distribution of the sampled point clouds P_S close to the distribution of the input point clouds P . The CD loss is expressed as follows:

$$L_{cd} = \frac{1}{|P_S|} \sum_{p \in P_S} \min_{p' \in P} \|p - p'\|_2^2 + \frac{1}{|P|} \sum_{p' \in P} \min_{p \in P_S} \|p' - p\|_2^2 \quad (17)$$

The repulsion loss L_{rep} prompts that the sampled point cloud P_S closes to the input point cloud P , while the sampled point p' is far away from other points around another sampled point. The repulsion loss L_{rep} is expressed as follows:

$$L_{rep} = -\frac{1}{|P_S|} \sum_{j=1}^{|P_S|} \sum_{j'=1}^{|P_S|} \min_{p' \in P, p'' \neq p'_j} \|p' - p''\|_2^2 \quad (18)$$

Accordingly, a joint loss L_{joint} is designed to train the network for adjusting the distribution of the sampled points, which can effectively ensure that the sampling results are evenly distributed in the overall and local regions.

$$L_{joint} = L_{cd} + L_{rep} \quad (19)$$

4. Experiments and results

To evaluate the performance of the proposed method, a series of experiments are conducted. In this section, we present the results of our approach to point cloud reconstruction and classification. The training dense point clouds include the Stanford Bunny and the real-world Terracotta Warrior Fragments. The classification task is benchmarked on the datasets of ModelNet, ScanObjectNN, and the Terracotta Warrior fragments. ModelNet is a synthetic dataset, while, both ScanObjectNN and the Terracotta Warrior fragments are real-world datasets.

4.1. Architecture

In TGPS-DGA-Net, the augmented data first goes through the NEC cell to extract local features with 16 dimensions, then pass through two consecutive 16-dimensional IOA layer and finally concatenate the output of each IOA layer. To enrich local relations between points, the feature with 512 dimensions is formed by DGA EConv. In the down sampled GPcS layer, the sampled ratio k is set to 256. Finally, the final output passes through two DGA EConv layers with dimensions of [40,41]. For classification, at the end of the network are two linear layers each followed by batch normalization and a LeakyReLU activation function with a negative slope of 0.2 while the final layer directly outputs predictions. In addition, for reconstruction, we use the same encoder architecture as in classification. The parameters of the decoder are followed by FoldingNet [11]. Our proposed method optimizes the joint loss separability to balance the distribution of the sampled points. When the weight is taken as 1, that is, $L_{\text{joint}} = L_{\text{cd}} + L_{\text{rep}}$ can make the point cloud distribution the most uniform.

The network is trained for 250 epochs on an NVIDIA GTX 1080Ti GPU and PyTorch v1.2, using Adam optimizer without weight decay. An initial learning rate of 0.01, and an initial momentum for Batch Normalization layers of 0.9 are set. The batch size is 24 for classification and 20 for reconstruction. The neighborhood value k is set to 16 for classification and 20 for reconstruction.

4.2. Reconstruction

The folding-based decoder is added to the DGA EConv for the reconstruction task. It should be noted that the dense point clouds of the Terracotta Warrior Fragments are scanned by trained students in our lab using a Creaform VIU handy scanner. Several dense point cloud models used are shown in Figure 5. To verify the validity of the sampled points, Geomagic software is used to obtain the triangular patch surface and fit the results of the point cloud simplification.

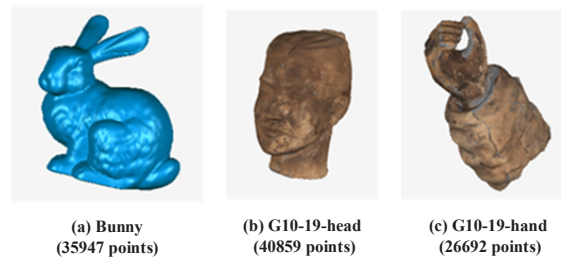


Fig. 5. Illustration of the representative models.

As shown in Figure 6 and Figure 7, the simplified results and triangular patch surface reconstruction results of the Bunny and the Terracotta fragments. In addition, the quantitative results are shown in Table 1.

The simplification rate of our TGPS and GDS [20] is usually $1/n^2$, while the method in [3] needs to extract feature points and simplify non-feature points separately. Therefore, there is no

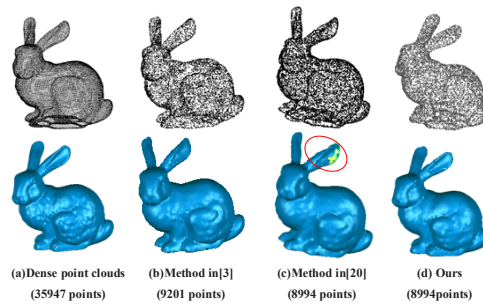


Fig. 6. Simplified and reconstruction results of Bunny

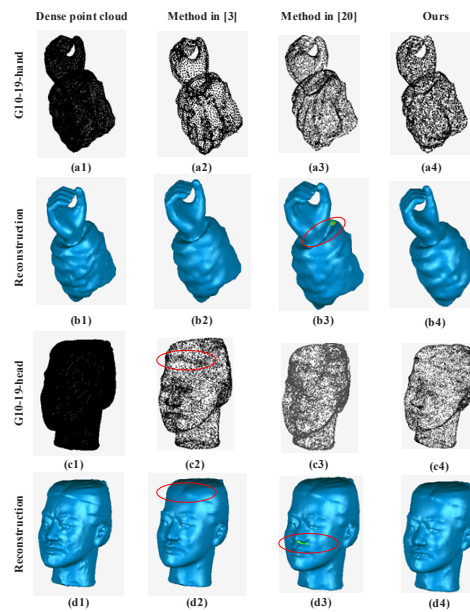


Fig. 7. Simplified and reconstruction results of G10-19-hand and G10-19-head.

Table 1. Comparison of maximum (average) errors of different methods. (10^{-3} mm)

Method	Method in [3]		Method in [20]		Ours	
	Δ_{\max}	Δ_{avg}	Δ_{\max}	Δ_{avg}	Δ_{\max}	Δ_{avg}
Bunny	0.3981	0.0736	0.3080	0.1143	0.2067	0.0488
G10-19- hand	0.5910	0.2885	0.7611	0.3488	0.4259	0.1537
G10-19- head	0.6307	0.2744	0.7004	0.3750	0.5083	0.2439

guarantee that the number is the same as our TGPS and GDS. For fairness, the number of [3] should be close to that of the other two simplified results. For Bunny, the number of original point clouds is 35974, and the number should be 8994 when the simplification rate is 25%, however the number of the method in [3] is 9201, approximately 25% of the simplification rate. As shown in Figure 6, the feature points of the ear in [20] are sparsely distributed, and there are holes in the surface reconstruction results. The point cloud simplification results and reconstruction results of our TGPS and the method in [3] both are similar to the original point clouds, and the simplified

results of our TGPS are obtained by network learning without manual intervention. In summary, our TGPS is more effective.

Figure 7 shows the different simplification and reconstruction results of G10-19-hand and G10-19-head. Figure 7 (a1) and (c1) are the dense point clouds of the G10-19-hand and G10-19-head, the number of points is 26692 and 40859, respectively. (a4) and (c4) are our simplification results with the number of points of 6773 and 10214; (a2) and (c2) are the simplification results of the method in [3], with the number of points of 6886 and 10103; (a3) and (c3) are the simplification results of GDS [20], with the same number as our TGPS; (b1-b4) and (d1-d4) are the triangular patch surface reconstruction results by Geomagic Wrap.

For the G10-19-hand, the simplification rates of the three methods all remain at about 25%. In Figure 7(b3), we can see holes in the wrist of the hand model. The method in [3] and our method can both obtain better surface reconstruction results, which indicates the simplified results are better. For the G10-19-head model, the feature points of the nose, eyes, mouth, and ears of the simplified model obtained by our TGPS can be well preserved, and the distribution of the feature points is relatively uniform in the relatively flat parts of the forehead, cheeks, and neck. However, the feature points of the hairline above the forehead of the simplified model obtained by the method in [3] are lost, which leads to the fact that the part is relatively flat and the feature outline is not obvious when it is converted into a triangular mesh model. In summary, our TGPS has the best simplification results and surface reconstruction results.

To evaluate the accuracy of the simplified point clouds, the geometric error between the original and simplified point clouds should be measured. The quantitative errors are shown in Table 3, where Δ_{\max} and Δ_{avg} represent the maximum error and average error of the geometry which are proposed by Shi et al. [42]. In Table 1, the method in [3] is the largest among the three methods in terms of both Δ_{\max} and Δ_{avg} , which is consistent with the experimental results in Figure 6 and Figure 7. The simplification results of our TGPS are significantly better than the method in [3] and [20], with minimal differences from the original point clouds while retaining a similar number of points, that is, the simplified results are closer to the original point clouds, which is beneficial for surface reconstruction. These quantitative results indicate that our TGPS comprehensively outperforms the existing simplified methods, and can better preserve the feature points of models.

4.3. Classification

In this section, to demonstrate the effectiveness and efficiency of the proposed TGPS-DGA-Net, we conducted experiments on three public datasets (ModelNet10, ModelNet40, ScanObjectNN) and a real-world dataset (Terracotta Warrior fragments). The results show that our proposed framework greatly outperforms all existing methods. We also perform a series of experiments to analyze the importance of each component in our method.

4.3.1. Datasets and implementation details

ModelNet: ModelNet40 contains 12,311 CAD models from 40 man-made object categories, of which 9,843 objects are for training and 2,468 objects are for testing. As a subset of ModelNet40, ModelNet10 contains 4899 models from 10 categories and is split into 3991 for training and 908 for testing. Each sample only retains 1024 uniformly distribute points as input, and only the coordinates (x, y, z) of the sampled points are used in the experiment.

ScanObjectNN: ScanObjectNN is a real-world point clouds object based on indoor scene scan data, containing 15 categories and a total of 2880 objects. The training set contains 2304 objects and the test set contains 576 objects.

Terracotta Warrior fragments: After the above simplification method, the simplified model of the Terracotta Warriors can be obtained. The current Terracotta Warrior fragments dataset was randomly sliced using the Blender software from 40 whole Terracotta Warriors, and finally

get 3250 fragments, which can be divided into four categories: (Arm: 800, Body: 810, Head: 810 and Legs: 830).

Figure 8 illustrates the randomly sliced fragments of the kc02f02-Arm model by Blender. As shown in Figure 8(b), these fragments vary in size and the number of points they contain. Hence, the obtained point clouds of the sliced fragments need to be down-sampled or interpolated, finally ensuring that each fragment has a fixed number of 2048. However, the number of the Terracotta Warrior fragments is not far from enough for training deep neural networks. To improve the robustness of the network, the sliced fragment point clouds are resampled into four non-overlapping point clouds, and the extended dataset is generated. Among them, 10144 patches for training (Arm:2656, Body:2720, Head:2272, Leg:2496) and remained 1852 for testing (testArm:476, testBody:504, testHead:428, testLeg:444). For training, we sample 1024 points and normalize them into a unit ball as input. Figure 9 shows the processed Terracotta Warrior fragments.

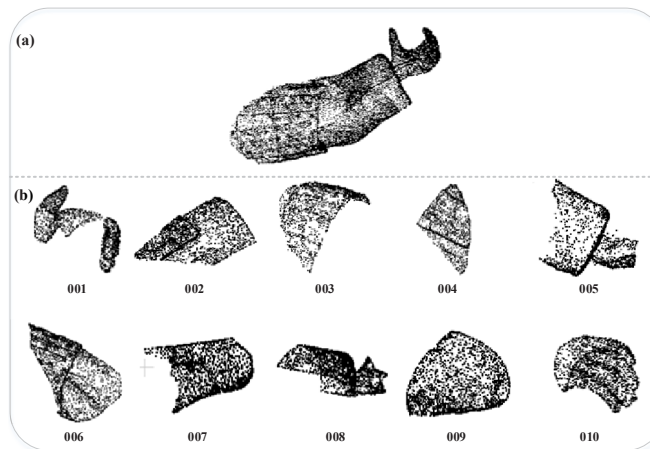


Fig. 8. Point cloud patches of kc02f02-Arm.

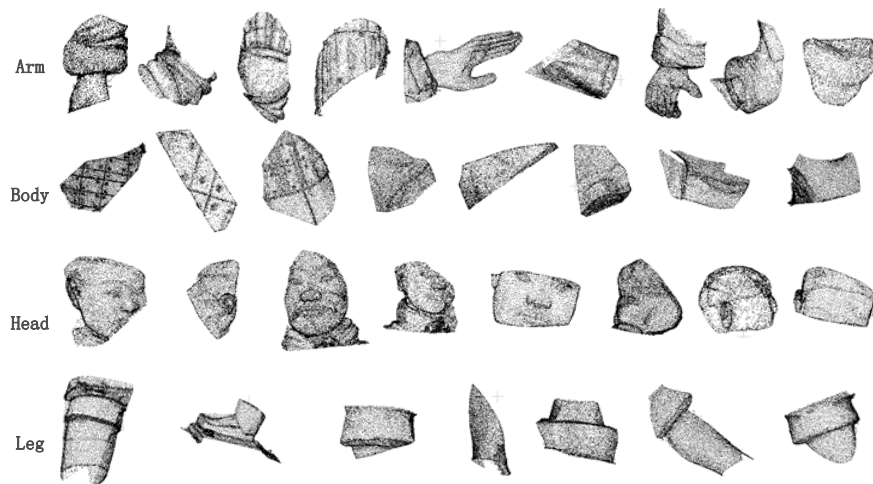


Fig. 9. Illustration of the processed Terracotta Warrior fragments.

Table 2. Comparisons of the classification accuracy (%). (“-” stands for unknown value)

Method	ModelNet40		ModelNet 10		ScanObjectNN	
	OA	CA	OA	CA	OA	CA
PointNet [8]	89.20	86.20				
PointNet++ [9]	90.70	-	-	-	82.30	
spiderCNN [14]	92.40	-	-	-	79.50	-
pointCNN [15]	92.50	88.80	-	-	86.10	-
DGCNN [12]	91.84	89.40	-	-	82.80	-
WCP-Net [19]	92.41	90.53	-	-	-	-
PST-NET [33]	89.20	-	-	-	-	-
S-Net [17]	89.20		-	-		-
SampleNet [18]	90.00	-	-	-	-	-
EC-GDP [20]	93.20	90.30	-	-	85.00	-
TGPS-DGA-Net	93.23	91.25	95.26	-	86.16	83.68
TGPS-EC-Net	91.94	-	94.71	94.38	85.14	82.47

4.3.2. Experiments on the public datasets

(1) Comparing with existing classification methods

In Table 2, the first four methods PointNet++ [9], SpiderCNN [14], PointCNN [15], and DGCNN [12] were general networks for point cloud learning, and the down-sampling method used was the FPS.

Table 2 shows the results of different methods evaluated regarding the overall accuracy(OA) and per-class mean accuracy(CA). In this highly competitive dataset, our model achieves the highest performance in terms of OA with 93.23%. Compared with PointNet++, our TGPS-DGA-Net applies the attention graph convolution to complete the construction of edge features and effectively improves the accuracy by 2.53%. Compared with DGCNN, our method outperforms it by 1.39%. As a model based on Transformers, our TGPS-DGA-Net is much higher than PST-NET by 4.03%. As a task-driven sampling network, our method outperforms by 3% than that of PST-NET, S-Net, and SampleNet. When the DGA EConv in our TGPS-DGA-Net is replaced by the general EdgeConv, named GTS-EC-Net, the result is reduced by 1.29%, which further illustrates the effectiveness of the proposed DGA EConv.

Table 2 also exhibits the results of the methods on the ScanObjectNN dataset. Our TGPS-DGA-Net with 86.16% outperformed Pointnet++ and DGCNN by 3.86% and 3.36%, respectively. This indicates that the TGPS-DGA-Net is more robust to the interference caused by the confusion between foreground and background points, and the reason is that the sampling method can select key points through the TGPS. Experiments demonstrate that our TGPS-DGA-Net achieves comparable performance with other approaches in point cloud classification, and also has good generalization ability for complex scenes.

(2) Comparing with existing sampling methods

To verify the feasibility and effectiveness of the proposed TGPS, we conduct two groups of experiments. The first group of experiments exhibited the accuracy of our TGPS in different sampling rates. From the last row in Table 3, we can see that the accuracy of our proposed method tends to increase first and then decrease as the sampling rate decreases. When the number of sampled points M is set to 256, the TGPS-DGA-Net achieves the best classification accuracy.

The offset of the accuracy is only 0.24% when M is 1024 and 64. Experiments further prove that the TGPS is still effective in retaining feature points with fewer sampled points.

Table 3. Classification results with different down-sampling rates(%)

Methods	1024(N)	512(N/2)	256(N/4)	128(N/8)	64(N/16)
RS	89.20	88.20	86.60	86.20	81.50
FPS [9]	89.20	88.30	88.10	87.90	86.10
Sample-Net [18]	89.20	87.80	88.30	88.60	87.70
PST-Net [33]	89.20	87.90	83.20	80.10	76.10
WCPL [19]	92.41	91.54	90.73	89.71	89.16
GDS [20]	92.90	93.20	93.20	92.40	93.20
TGPS	92.06	92.50	93.23	91.98	91.82

The second group of experiments is comparing the proposed TGPS with the previous methods. For fairness, all the reported methods are given 512 points. In Table 2, our TGPS achieves the best sampling results. Among them, FPS and RS are simple and the accuracy is far inferior to our TGPS. Sample-Net and PST-Net are task-driven sampling methods, but the classification accuracy needs to be improved. Our TGPS obtains up to 2.5% and 0.03% improvement over WCPL and GDS, respectively. Furthermore, Figure 10 shows the simplified results of WCPL, GDS, and our TGPS. The number of all sampled points is 256. From the comparison results, we can see that the sampled points obtained by our TGPS are evenly distributed. At the same time, feature points and contour points are effectively preserved.

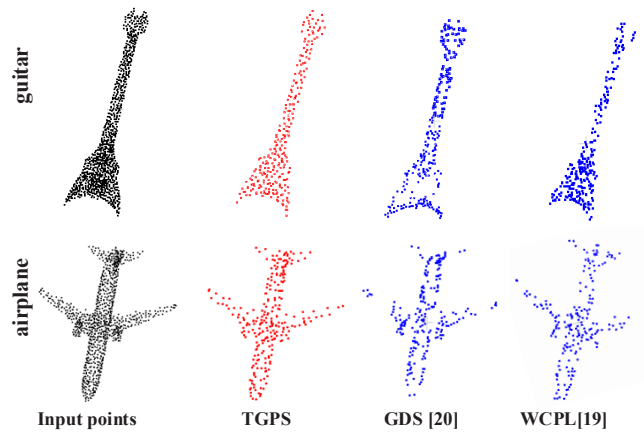


Fig. 10. Visualization results of the sampling methods.

4.3.3. Experiments on Terracotta Warrior dataset

We benchmark our TGPS-DGA-Net on 3D Terracotta Warrior fragments and compare the performance with the existing methods. As shown in Table 4, our method can achieve the highest accuracy. Compared with the traditional method [43], our method outperforms by 10.04%. Moreover, our method outperforms the method in [4] by 6.27%, which included both XYZ coordinates and RGB information. Compared with the method [5] uses richer feature information (with normal vectors) as inputs, our method has an improvement of 1.46%. The implementation results further demonstrate the effectiveness of the GDA EConv in extracting local features.

Table 4. Compared with other methods on the 3D Terracotta Warrior fragment dataset. (“P” stands for Xyz-coordinates; “N” stands for normal vector; “I” stands for RGB information.)

Method	Data type	Deep model	Supervised	Accuracy(%)
Method in [43]	P	False	True	87.64
PointNet [8]	P	True	True	88.93
Method in [4]	P, I	True	True	91.41
AMS-Net [5]	P	True	True	95.68
AMS-Net [5]	P,N	True	True	96.22
UMA-Net [6]	P	True	False	93.90
Ours	P	True	True	97.68

4.4. Sensitivity analysis

To further evaluate the importance of each component of the proposed TGPS-DGA-Net, we perform ablation experiments on the ModelNet40.

4.4.1. Effect of feature dimension

The effect of the feature dimension of the output feature vector on the accuracy is shown in Table 5. The TGPS does not achieve the best performance whether the maximum value of 1024 or the minimum of 128 is taken, but our TGPS reaches its best at an intermediate feature dimension of 256. Even with a dimension size of 128, the accuracy of the proposed TGPS (91.90%) is still higher than that of the PointNet with a dimension size of 1024 (89.20%) listed in Table 1.

Table 5. The effect of different feature dimensions (%)

N-dim	1024	512	256	128
Accuracy	92.50	92.30	93.23	91.90

4.4.2. Effect of relational functions

Experiments are conducted to evaluate the effectiveness of different relationship functions with 256 sampled points, and the comparison results are shown in Table 6. The Concatenation achieves the highest accuracy, slightly higher than the Summation by 0.52%. While the Subtraction is 1.41% lower than the Concatenation. The simplified point cloud should maintain similarity to the original model, but the Subtraction may result in an increased loss of contextual information. Instead, the Connection is aggregated with the input features and attention features, which the network more robust. The Concatenation outperforms the Hadamard product and Dot product by 2.02% and 20.54%, respectively. Therefore, all experiments in Sec.3.1 and Sec.3.2 adopt the self-attention structure based on the Concatenation.

Table 6. The effect of different relational functions (%)

Self-attention	Accuracy
Concatenation	93.23
Summation	92.71
Subtraction	91.82
Hadamard product	91.21
Dot product	72.69

4.4.3. Effect of edge feature

As a core cell for extracting local information, DGA EConv constructs the nearest neighbor graph for the sampled points. In this group of experiments, different combinations of the center point, the nearest neighbor point, and the Euclidean distance information are used to represent the edge features. Among them, the combination of different features is achieved through splicing operation. As shown in Table 7, Model B has the best results. The reason is that the edge features of model A only considers global features, while model C decreases in accuracy due to information redundancy. While, model B includes both the local and global features of the model. Experiments show that the proposed DGA EConv can effectively retain the key points and important features related to the downstream classification task, thus improving the network performance.

Table 7. The effect of different edge features (%)

Model	Edge features			Accuracy
	p_i	$p_i - p_{ij}$	$\ p_i - p_{ij}\ _2$	
A	√			91.90
B	√	√		93.23
C	√	√	√	92.86

4.4.4. Effect of the number of IOA module layers

To further verify the effectiveness of the number of layers of IOA modules included in the Transformer unit on feature extraction, 1-layer, 2-layer, and 4-layer IOA modules are used for comparison on the verification set, and the results are shown in Table 8. The results show that when the number of layers is 1, the accuracy is 91.77%, which is the lowest among the three forms. When the number of layers is 2, the accuracy is significantly improved by 1.46%. When the number of layers is 4, the result is closed to that at 2. Considering more IOA layers will lead to an increase in model complexity, 2-layer IOA is the best choice.

Table 8. The effect of different numbers of self-attention layers (%)

N-dim	1	2	4
Accuracy	91.77	93.23	93.25

4.4.5. Effect of the loss function

To verify the effect of different loss functions on classification results, Table 9 shows the classification accuracy. As suggested in Table 9, the result of the joint loss L_{joint} can achieve the best performance. It is further proved that our joint loss helps to extract the important points for classification.

Table 9. The effect of the loss function (%)

Loss	CD-only (L_{cd})	Rep-only (L_{rep})	CD + Rep (L_{joint})
Accuracy	91.98	92.22	93.23

In addition, as exhibited in Figure 11, the sampled points of the CD-only loss are relatively concentrated on the contour of the 3D object, and the result of Rep-only loss is relatively even than that with CD-only loss, there are still more obvious holes in the lower right corner of the guitar and in the fuselage area of the aircraft. the result of the total loss achieves the best

performance. When adopting the joint loss L_{joint} , the distribution of the sampled points is more uniform in the overall and local regions, and the contour features are well preserved.

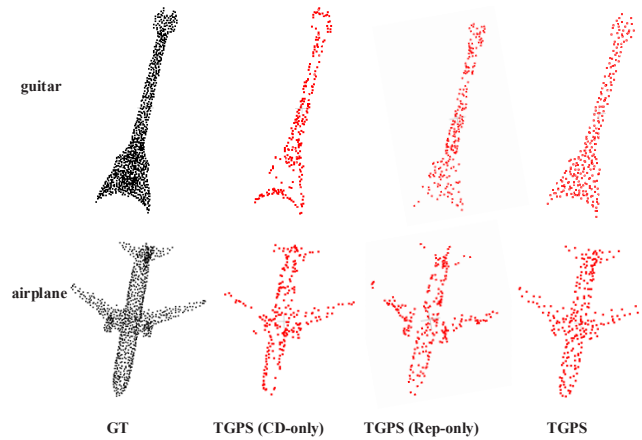


Fig. 11. Results of different losses.

5. Conclusion

Due to the dense point clouds of the cultural relic collected by the 3D scanner requiring excessive storage and time costs, we propose an end-to-end point cloud simplification method for the Terracotta Warrior fragments. In this paper, we propose a learnable point cloud down-sampling method, which uses the importance of each point to perform non-repetitive point down-sampling under the guidance of global features, and retains important points related to downstream tasks. Among them, the TGPS can retain important point features and their coordinates by training the network. In addition, to further learn rich local representation, the DGA EConv proposed to perform local feature aggregation for the neighborhood graph. Experiments demonstrate that our TGPS can effectively simplify the models and capture more detailed features according to the point cloud classification and reconstruction tasks.

However, there are limitations that our method is relatively sensitive to outliers or noise. In the future, we could further propose a differentiable adaptive sampling method to readjust the spatial distribution of sampled points in data-driven manners to further improve the robustness of the network. Furthermore, combined with knowledge distillation, we will propose an efficient and lightweight model that can achieve network acceleration.

Funding. National Natural Science Foundation of China (61701403, 61731015, 62271393); National Key Research and Development Program of China (2019YFC1521102, 2019YFC1521103, 2020YFC1523301); Key R&D Projects in Qinghai Province (2020-SF-140); Key Research and Development Program of Shaanxi Province (2019GY215, 2021ZDLSF06-04); China Postdoctoral Science Foundation (2018M643719); Shaanxi Provincial Department of Education Special Project (19JK0842).

Acknowledgments. We thank Emperor Qinshihuang's Mausoleum Site Museum for providing the Terracotta Warriors data.

Disclosures. The authors declare no conflicts of interest.

Data availability. The Terracotta Warriors Data underlying the results presented in this paper are not publicly available at this time but may be obtained from the authors upon request.

References

1. X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 1907–1915.

2. Q. Wang, S. Liu, J. Chanussot, and X. Li, "Scene Classification With Recurrent Attention of VHR Remote Sensing Images," *IEEE Trans. Geosci. Electron.* **57**(2), 1155–1167 (2019).
3. G. Geng, J. Liu, X. Cao, Y. Liu, W. Zhou, F. Zhao, L. Su, K. Li, and M. Zhou, "Simplification method for 3D Terracotta Warrior fragments based on local structure and deep neural networks," *J. Opt. Soc. Am. A* **37**(11), 1711 (2020).
4. K. Yang, X. Cao, G. Geng, K. Li, and M. Zhou, "Classification of 3D terracotta warriors fragments based on geospatial and texture information," *J. Visualization* **24**(2), 251–259 (2021).
5. J. Liu, X. Cao, P. Zhang, X. Xu, Y. Liu, G. Geng, F. Zhao, K. Li, and M. Zhou, "AMS-Net: An Attention-Based Multi-Scale Network for Classification of 3D Terracotta Warrior Fragments," *Remote Sens.* **13**(18), 3713 (2021).
6. J. Liu, Y. Tian, G. Geng, H. Wang, D. Song, K. Li, M. Zhou, and X. Cao, "UMA-Net: an unsupervised representation learning network for 3D point cloud classification," *J. Opt. Soc. Am. A* **39**(6), 1085 (2022).
7. W. Yao, T. Chu, W. Tang, J. Wang, X. Cao, F. Zhao, K. Li, G. Geng, and M. Zhou, "SPPD: A Novel Reassembly Method for 3D Terracotta Warrior Fragments Based on Fracture Surface Information," *IJGI* **10**(8), 525 (2021).
8. R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2017), pp. 77–85.
9. C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2017), 30.
10. J. Li, B. M. Chen, and G. H. Lee, "SO-Net: Self-Organizing Network for Point Cloud Analysis," in (2018), pp. 9397–9406.
11. Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation," in (2018), pp. 206–215.
12. Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic Graph CNN for Learning on Point Clouds," *ACM Trans. Graph.* **38**(5), 1–12 (2019).
13. L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph Attention Convolution for Point Cloud Semantic Segmentation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2019), pp. 10288–10297.
14. Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters," in (2018), pp. 87–102.
15. Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution On X-Transformed Points," in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2018), 31.
16. J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian, "Modeling Point Clouds With Self-Attention and Gumbel Subset Sampling," in (2019), pp. 3323–3332.
17. O. Dovrat, I. Lang, and S. Avidan, "Learning to Sample," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), pp. 2755–2764.
18. I. Lang, A. Manor, and S. Avidan, "SampleNet: Differentiable Point Cloud Sampling," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2020), pp. 7575–7585.
19. E. Nezhadarya, E. Taghavi, R. Razani, B. Liu, and J. Luo, "Adaptive Hierarchical Down-Sampling for Point Cloud Classification," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2020), pp. 12953–12961.
20. J. Wang, Y. Zhao, T. Liu, and S. Wei, "GDS: Global Description Guided Down-Sampling for 3D Point Cloud Classification," in *Proceedings of the 2020 4th International Conference on Vision, Image and Signal Processing* (ACM, 2020), pp. 1–6.
21. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," (2019).
22. A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," 12 (n.d.).
23. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16 × 16 Words: Transformers for Image Recognition at Scale," (2021).
24. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2017), 30.
25. N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," (2020).
26. P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-Alone Self-Attention in Vision Models," in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2019), 32.
27. J. Dai, "Deformable DETR: Deformable Transformers for End-to-End Object Detection," 26 (n.d.).
28. H. Zhao, L. Jiang, J. Jia, P. H. S. Torr, and V. Koltun, "Point Transformer," in (2021), pp. 16259–16268.
29. M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "PCT: Point cloud transformer," *Comput. Vis. Media* **7**(2), 187–199 (2021).
30. I. Misra, R. Girdhar, and A. Joulin, "An End-to-End Transformer Model for 3D Object Detection," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (IEEE, 2021), pp. 2886–2897.
31. X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3D Object Detection with Pointformer," (2021).

32. C. Zhou, Z. Luo, Y. Luo, T. Liu, L. Pan, Z. Cai, H. Zhao, and S. Lu, "PTTR: Relational 3D Point Cloud Object Tracking With Transformer," in (2022), pp. 8531–8540.
33. X. Wang, Y. Jin, Y. Cen, C. Lang, and Y. Li, "PST-NET: Point Cloud Sampling via Point-Based Transformer," in *Image and Graphics*, 12890 Y. Peng, S.-M. Hu, M. Gabbouj, K. Zhou, M. Elad, and K. Xu, eds., Lecture Notes in Computer Science (Springer International Publishing, 2021), 12890, pp. 57–69.
34. C. Wang, B. Samari, and K. Siddiqi, "Local Spectral Graph Convolution for Point Set Feature Learning," in (2018), pp. 52–66.
35. J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral Networks and Locally Connected Networks on Graphs," (2014).
36. G. Te, W. Hu, Z. Guo, and A. Zheng, "RGCNN: Regularized Graph CNN for Point Cloud Segmentation," (2018).
37. M. Simonovsky and N. Komodakis, "Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs," in (2017), pp. 3693–3702.
38. K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, "Linked Dynamic Graph CNN: Learning on Point Cloud via Linking Hierarchical Features," (2019).
39. X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local Neural Networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 7794–7803.
40. B.-Q. Shi, J. Liang, and Q. Liu, "Adaptive simplification of point cloud using -means clustering," *Comput.-Aided Des.* **43**(8), 910–922 (2011).
41. G. Du, M. Zhou, C. Yin, Z. Wu, and W. Shui, "Classifying fragments of terracotta warriors using template-based partial matching," *Multimed. Tools Appl.* **77**(15), 19171–19191 (2018).