Research Article Vol. 39, No. 6 / June 2022 / Journal of the Optical Society of America A 1085

Journal of the Optical Society of America A
OPTICS, IMAGE SCIENCE, AND VISION

UMA-Net: an unsupervised representation learning network for 3D point cloud classification

Jie Liu,[1,2,†] Yu Tian,[1,2,†] Guohua Geng,[1,2,3,†] Haolin Wang,[1,2] Da Song,[1,2] Kang Li,[1,2] Mingquan Zhou,[1,2] and Xin Cao[1,2,4]

[1]School of Information Science and Technology, Northwest University, Xi'an, Shaanxi 710127, China
[2]National and Local Joint Engineering Research Center for Cultural Heritage Digitization, Xi'an, Shaanxi 710127, China
[3]e-mail: ghgeng@nwu.edu.cn
[4]e-mail: xin_cao@163.com

Received 14 February 2022; revised 18 April 2022; accepted 7 May 2022; posted 11 May 2022; published 26 May 2022

The success of deep neural networks usually relies on massive amounts of manually labeled data, which is both expensive and difficult to obtain in many real-world datasets. In this paper, a novel unsupervised representation learning network, UMA-Net, is proposed for the downstream 3D object classification. First, the multi-scale shell-based encoder is proposed, which is able to extract the local features from different scales in a simple yet effective manner. Second, an improved angular loss is presented to get a good metric for measuring the similarity between local features and global representations. Subsequently, the self-reconstruction loss is introduced to ensure the global representations do not deviate from the input data. Additionally, the output point clouds are generated by the proposed cross-dim-based decoder. Finally, a linear classifier is trained using the global representations obtained from the pre-trained model. Furthermore, the performance of this model is evaluated on ModelNet40 and applied to the real-world 3D Terracotta Warriors fragments dataset. Experimental results demonstrate that our model achieves comparable performance and narrows the gap between unsupervised and supervised learning approaches in downstream object classification tasks. Moreover, it is the first attempt to apply the unsupervised representation learning for 3D Terracotta Warriors fragments. We hope this success can provide a new avenue for the virtual protection of cultural relics.

https://doi.org/10.1364/JOSAA.456153

1. INTRODUCTION

Recently, as an effective representation for the 3D object, point clouds have attracted noticeable attention. The rapid development of laser scanner technology makes the acquisition of point clouds more simple and convenient. Furthermore, point clouds are widely used for autonomous driving [1], human–computer interactions [2], and robotics [3]. The great success in point cloud classification, part segmentation, and object detection applying deep neural networks has inspired us to explore their comprehensive capabilities in a wide variety of computer vision tasks. Currently, PointNet is used as the dominant framework for point cloud classification and segmentation [4], while the main limitation of PointNet is the only dependent on the max-pooling layer to learn global representations. Although PointNet fails to capture local structures, a great number of variants models [5–10] based on PointNet have achieved suitable performance, which obtains local features by constructing neighborhood information to make up for the shortcomings of PointNet. Wang *et al.* [6] present a spectral graph convolution on a local graph to fully exploit the neighboring points' relative structure and features. The method requires no pre-computation of the graph Laplacian matrix and graph coarsening hierarchy. PointWeb [8] explores the relations between all pairs of points in a local region by densely constructing a locally fully linked web.

Unfortunately, during the process of training the above-mentioned deep neural networks, large amounts of labeled data are typically required to learn sufficient representations for 3D scene understanding tasks [11], especially for a real-world dataset, for example, the Terracotta Warriors fragments. Recently, a considerable number of such tasks have been proposed, such as predicting image rotations [12], image inpainting [13], solving jigsaw puzzles made from image patches [14], etc. Furthermore, the unsupervised learning methods for image feature learning have obtained great success. In recent studies, there is an increased interest in learning representations in an unsupervised manner, which addresses point cloud understanding tasks with insufficient labeled data. Among the existing unsupervised learning methods on point clouds, autoencoders (AE) [10,15–18], and generative adversarial networks (GAN) [19,20] are considered representative models. *l*-GAN [16] simply produces the point clouds through multilayer perceptrons (MLPs) from the codewords, and FoldingNet [10] is a kind of

novel AE model that directly folds the entire point set based on the duplicated global feature instead of the fully connected decoder. The 3D point cloud data is reconstructed by training the network, from which the features are obtained, while these methods are only effective in obtaining structural and low-level information from point clouds. Liu *et al.* [17] proposed a local-to-global autoencoder (L2G-AE) to simultaneously learn the local and global structure of point clouds through local-to-global reconstruction for point cloud understanding in the shape classification. Additionally, several methods cluster the 3D point cloud features obtained by backbone into different clusters [21–23]. The overall idea of these methods is very simple, and the clustering algorithm brings with it the drawbacks of computational complexity when dealing with point cloud data. Accordingly, how to make the global feature more representative of the original point cloud still is a key and difficult issue. Meanwhile, the quality of the representative directly affects the downstream tasks, for example, 3D object classification, segmentation, and detection.

With the transformation of traditional museums into digital museums, scarce cultural relics can be well exchanged and shared, and the display of cultural relics will become rich and diverse. As one of the great discoveries in the history of archaeology, Terracotta Warriors have been predominantly found in fragments due to the natural environment and human factors. The traditional approaches [24–26] are primarily based on profile features, texture features, or multi-features, which require the design of accurate feature description operators from experts and a lot of time. In recent years, some deep learning methods are also applied to the restoration of cultural relics, such as cultural relic fragments simplification [27], cultural relic fragment classification, and segmentation [28,29]. Yang *et al.* [28] proposed a novel method that can directly analyze the point cloud and texture image of the fragment and output its category. Based on the multi-scale and self-attention strategy, Liu *et al.* [29] presented a novel hierarchical network to enhance the capability of extracting both low-level and high-level features of the 3D Terracotta Warrior fragments. However, representation learning in the above methods relies on a large amount of labeled data for training. At present, there are few studies based on unsupervised representation learning for 3D cultural relics.

Motivated by the above analysis, we propose an unsupervised representation learning network based on multi-scale feature aggregation, named UMA-Net, for the downstream point cloud classification task. First, with the inspiration of the multi-scale strategy and shellconv in [9], we propose a novel encoder based on a multi-scale shell block to extract the local features from different scales hierarchically in a simple yet effective manner. Subsequently, the encoder further aggregates all the information extracted from the point cloud into global representations G (see in Fig. 1). Second, the cross-dim-based decoder is employed to decode the learned global representations into 3D coordinates in a coarse-to-fine way, which can get a better-reconstructed point cloud. Third, with the purpose of getting a good metric for measuring the similarity between local features and global representations, a compound loss is presented, which takes angle relationship and distance into account. Finally, a linear classifier is trained by using the global representations obtained from the pre-trained model. Additionally, our model

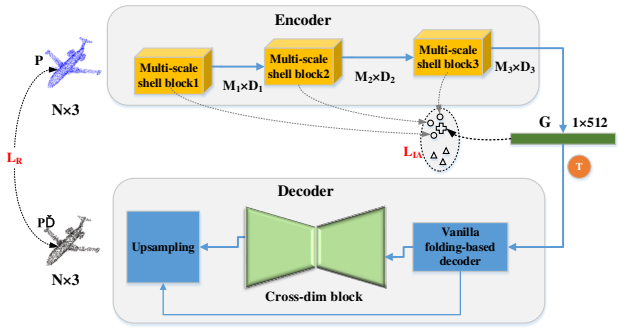

Fig. 1. Overview of our UMA-Net architecture. T, Tile; +, the anchor; o, the positive sample; △, the negative sample.

is evaluated on the public dataset (ModelNet40) and the real-world dataset (3D Terracotta Warriors fragments). To our knowledge, it is the first attempt to apply the unsupervised representation learning for 3D Terracotta Warriors fragments. Experimental results demonstrate that our UMA-Net achieves comparable performance with other approaches in point cloud classification. The main contributions of this work are summarized as follows:

- We propose UMA-Net as a novel deep learning model to perform unsupervised representation learning for the downstream 3D object classification, which is also the first attempt to apply the unsupervised representation learning for 3D Terracotta Warriors fragments.
- We introduce a hierarchical multi-scale shell-based encoder to improve the local feature expression ability of the model. Meanwhile, a cross-dim-based decoder is proposed to improve the quality of point cloud generation.
- We present an improved angular loss to get a good metric for measuring the similarity between local features and global representations.
- We demonstrate the effectiveness of the learned representations: our model achieves comparable results w.r.t prior unsupervised models and narrows the gap between unsupervised and supervised models in a downstream point cloud classification task.

The remainder of this paper is organized as follows. The overview of the proposed network and its sub-modules are given in Section 2. In Section 3, a benchmark dataset and a real-world dataset are introduced, and the experimental results are presented. Finally, the conclusions and future works are illustrated in Section 4.

2. METHODS

Our goal is to learn rich representations $G = \mu_\theta(p_i)$ in an unsupervised manner, in which μ_θ denotes a deep neural network with parameters θ, mapping point cloud p_i to global features G. To ensure that the global features G are equipped with the same semantic and structural information as the raw point cloud P, we propose to learn features by training networks to accomplish both predicting global features from local features and the reconstruction pretext tasks. The pipeline of our framework is illustrated in Section 2.A, which includes three main modules,

namely, the multi-scale shell block (Section 2.B), the cross-dim block (Section 2.C), and a novel similarity measure between local features and global features (Section 2.D). In the following subsections, each cell in the pipeline is introduced in detail.

A. UMA-Net Architecture

Given an unordered sparse point set $P = \{p_i \in \mathbb{R}^3, i = 1, 2, \cdots, N\}$ with N ($N = 1024$) points, each point in the point set is composed of a 3D coordinate (x, y, z). The main components of the proposed network are multi-scale shell block, cross-dim block, and improved angular loss in Fig. 1. Additionally, the encoder trained in the network can be used as pre-training for object classification tasks.

In the pre-train processing, first, the input P is sent to a hierarchical structure composed of three multi-scale shell blocks. In the encoder, we build a hierarchical grouping of points and progressively expand the receptive field hierarchy. The local features from the lth level F^l and the global features G with 512 dim are obtained. To ensure that a richer representation of the point cloud can be obtained by the global feature G, similarity metrics between local features and global features are established, which can be regarded as a self-supervised metric learning problem. The global features of the current object are defined as the positive example, and the global features of other objects are defined as the negative examples. Second, different from the previous deep metric learning (DML) methods that formulate objectives based on distance, our improved angular loss aims at constraining the angle at the negative point of triplet triangles. Combined with the angular loss, more structural information and semantic information are contained in the global feature. Third, DML can only guarantee that the local features are close to the global features, while the global features are not guaranteed to be representative, which is crucial for the unsupervised downstream task of point clouds. To ensure that the global features do not deviate from the input data, an auxiliary task (self-reconstruction) is proposed to enable the global features to capture the more basic structural information of the point cloud. In the decoder, the advantages of fully connected decoders and folding-based decoders are combined. Finally, after pre-training the model, the weights of the model are frozen, and the global features are extracted without any fine-tuning, a linear classifier is trained on them, and the classification accuracy is reported.

B. Multi-scale Shell Block

As one step of point cloud classification or segmentation processing, the performance of feature extraction is crucial. In this paper, with the inspiration from [9], the shell operator combined with a multi-scale strategy is proposed as the core cell of the encoder. The multi-scale shell block is illustrated in Fig. 2.

The structure of the shell block is formed by three layers: the sampling layer [see Fig. 2(a)], the building multi-scale spherical shell layer [see Fig. 2(b)], and the shell feature extraction layer [see Fig. 2(c)]. M centroid points $P_c^m = \{p_c^m \in \mathbb{R}^3, m = 1, 2, \cdots, M\}$ are selected from the input set P by the farthest point sampling (FPS) method in the sampling layer ($M < N$). For each sampled point, the

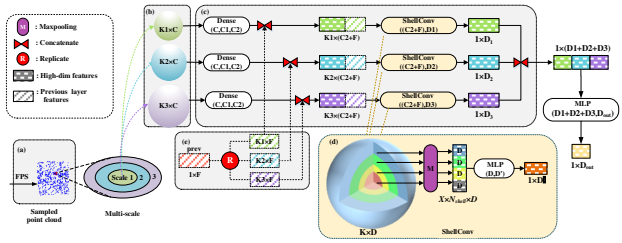

Fig. 2. Illustration of the multi-scale shell block.

K-nearest-neighbor (kNN) searching is employed to extract the features of all neighbor points. The proposed encoder takes three different scales as input, each of them containing $K_i (i = 1, 2, 3)$ points. One scale block is introduced in detail to easier understand as follows.

In the building spherical shell layer, first, we compute the distance between the neighbors and the centroid point. Then the distances are ordered, and the neighbors are distributed into different shells according to the distance. Each sphere is divided into different colors [see the left of Fig. 2(d)]. The number of points in each shell is fixed, and the number of points contained in each shell can reach a threshold value N_{shell}. Subsequently, the sphere continues to expand outward, until the next shell contains another N_{shell} points, and so on. If the local neighborhood of the centroid point is composed of X shells, the number of neighbors N_{nei} is defined as $X \times N_{\text{shell}}$ (here $K = N_{\text{nei}}$ in each scale). Finally, points with the size $M \times N_{\text{nei}} \times 3$ are obtained in the scale area of local regions.

In the shell feature extraction layer, the input of the convolution operation is composed of $(N_{\text{nei}} + 1)$ points. It is not practical to assign a convolution weight $W(q)$ to each neighbor point q, because the points are unordered. In this paper, the features of the points within the shell are integrated into the layer and assigned the same convolution weight to the features of the points in the same shell. The new convolution is defined as

$$F(p)^{(l)} = \sum_{X \in B_p^{(l)}} W_X^{(l)} F(X)^{(l-1)}, \tag{1}$$

where the superscript (l) denotes layer l and $F(\cdot)^{(l-1)}$ stands for features from the previous layer as shown in Fig. 2(e). B_p is a neighbors domain composed of multiple concentric spherical shells.

The order of the shells is from inner to outer; however, the points in each shell are still unordered. With the inspiration from [4], the max-pooling is introduced to aggregate features of the points in each shell in Eq. (2). Subsequently, 1D convolution is used to integrate the features of all shells to obtain the features for the centroid point as Eq. (3).

$$F(X) = \max_{q \in B_X} F(q), \tag{2}$$

where B_X denotes the region of a shell s.

$$F_q = \xi(C(F(X_i)), i = 1, 2, \ldots, s), \tag{3}$$

where C denotes concatenation operation, and ξ denotes 1D convolution.

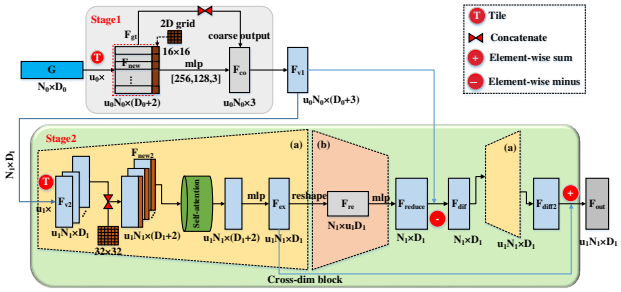

Fig. 3. Illustration of the coarse-to-fine decoder.

C. Coarse-to-Fine Decoder

The decoder is present to generate the output point cloud from the global features G. The point clouds with fully connected layers generated usually ignore geometric details in the local regions. The proposed decoder involves a combination of the advantages of fully connected decoders [16] and folding-based decoders [10] in a coarse-to-fine way to solve the above shortcomings. We divide the generation of the output point cloud into two stages. In Stage1, a coarse output F_{co} is generated by passing G through a vanilla folding-based decoder u_0 times and reshaping the output into a three-dimensional matrix (see the top of Fig. 3). In Stage2, we construct the cross-dim block to expand the point features (see the bottom of Fig. 3) and generate more consistent geometric details on the local regions. The detailed structure of the proposed decoder is illustrated in Fig. 3.

1. Vanilla Folding-Based Decoder

The folding-based decoder is good at approximating a smooth surface that represents the local geometry of a shape [10]. Inspired by this idea, we propose the vanilla folding-based decoder, which aims to fold the 2D grid into 3D space. Although Stage1 seems simpler than the cross-dim block, it provides Stage2 with a more expressive input. The detailed steps of the vanilla folding-based decoder are presented in Algorithm 1.

2. Cross-Dim Block

In this work, we first expand the features to generate F_{ex} and reduce them back, and then wecalculate the difference (F_{diff}) between the features before the Expanded unit [see Fig. 3(a)] and after the Reduced unit [see Fig. 3(b)]. We expand the difference value to F_{diff2}, and then we add F_{diff2} to F_{ex} to self-correct the expanded features. The detailed steps of the Cross-Dim Operator are presented in Algorithm 2.

Algorithm 1. Vanilla Folding-Based Operator

Input: $p \in P, u_0$ *Input point, the expand ratio.
Output: $F_{v1}(p)$ *Output features of cross-dim block of p.
1: $F_{gt}(p) \leftarrow$ tile $(G(p),\ u_0)$ *Tile G for u_0 times.
2: $F_{new}(p) \leftarrow$ con $(F_{gt}(p)$, 2Dgrids) *Concatenate F_{gt} with the 2D grid (16×16).
3: $F_{co}(p) \leftarrow$ reshape(mlp $(F_{new}(p))$) *Reshaped into a coarse output.
4: $F_{v1}(p) \leftarrow$ con $(F_{co}(p),\ F_{gt}(p))$ *Concatenate F_{gt} with the 2D grid.
5: **return** $F_{v1}(p)$

Algorithm 2. Cross-Dim Operator

Input: $p \in P, u_1$ *Input point, the expand ratio.
Output: $F_{out}(p)$ *Output features of cross-dim block of p.
Expanded unit *F_{v1} and F_{ex} as the input and output of the expanded unit, expand the point features u_1 times.
1: $F_{v2}(p) \leftarrow$ tile $(F_{v1}(p),\ u_1)$ *Tile F_{v1} for u_1 times.
2: $F_{new2}(p) \leftarrow$ con $(F_{v2}(p)$, 2Dgrids) *Concatenate F_{v2} with the 2D grid (32×32)
3: $F_{ex}(p) \leftarrow$ mlp (selfattention$(F_{new2}(p))$) *A self-attention module with MLPs to establish the inner links between features of F_{new2}.
Reduced unit *Reduce the expanded features.
4: $F_{re}(p) \leftarrow$ reshape(mlp $(F_{ex}(p))$) *Reshaped F_{ex} obtained from the expanded unit.
5: $F_{reduce}(p) \leftarrow$ mlp $(F_{re}(p))$ *Passed through a set of MLPs to make it into the original features.
6: $F_{diff}(p) \leftarrow F_{v1}(p) - F_{reduce}(p)$ *Compute the difference between the features before the Expanded unit and after the Reduced unit.
7: $F_{diff2}(p) \leftarrow$ expand unit$(F_{diff}(p))$ *Expand the difference value F_{diff} to F_{diff2}.
8: $F_{out}(p) \leftarrow F_{ex}(p) + F_{diff2}(p)$ *Add F_{diff2} to F_{ex}.
9: **return** $F_{out}(p)$

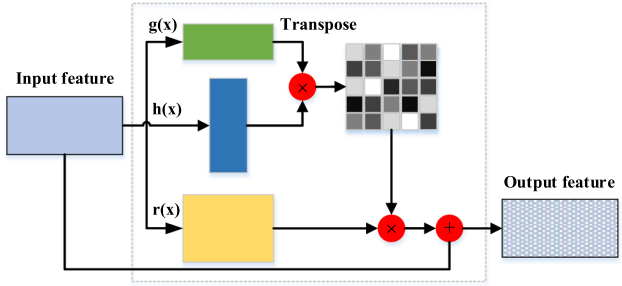

Fig. 4. Illustration of self-attention.

3. Self-Attention Unit

The self-attention module is used to establish the inner links between features so that similar features can be selectively aggregated together. The overall process of global features aggregated based on general self-attention is illustrated in Fig. 4. Given the ith point feature F_i, the output feature can be defined as

$$y_i = \sum_{x_i \in X} \theta(\sigma(\theta(g(F_i)^T, h(F_j))), \gamma(F_j)), \tag{4}$$

where $g(\cdot)$, $h(\cdot)$, and $\gamma(\cdot)$ denote the point-wise feature transformations; for simplicity, they are considered in the form of linear functions. θ is defined as the matrix (relation) function, and $\sigma(\cdot)$ is defined as a normalization function, such as softmax. We generate the output features, which are the sum of the input features and the weighted features.

D. Loss

Deep metric learning aims to learn embeddings that capture semantic similarity information among points, and the loss function is the key to DML success. In existing studies, the previous loss functions include the contrastive loss [30], the triplet loss [31,32], and some of its variants [33–35]. In this paper, a novel joint loss function is proposed to get a good metric for measuring the similarity between local features and global representations. The proposed loss function is described in detail as follows.

1. Improved Angular Loss

Each triplet $\{p_a, p_p, p_n\}$ consists of an anchor point p_a, a positive sample p_p, and a negative sample p_n in the iteration of a batch. The mapping function $f(\cdot)$ can map point p_i to an embedding feature vector $f(p_i) \in \mathbb{R}^d$. For convenience, $\psi_a = f(p_a), \psi_p = f(p_p), \psi_n = f(p_n)$. The goal of the triplet loss [32] is to pull the anchor point closer to the positive sample than to the negative sample by a fixed margin $\eta = 0$. The triplet loss can be formed as follows:

$$L_{\text{tri}} = \max(0,\ S(\psi_a, \psi_p) - S(\psi_a, \psi_n) + \eta), \tag{5}$$

where $S(m, n) = \|m - n\|_2^2$ denotes the Euclidean distance between m and n.

The triplet loss only focuses on the information of one negative sample in each optimization. To reduce the training burden while making full use of each batch of training samples, N-pair-mc loss [35] is introduced to identify one positive sample from (N-1) negative samples. The corresponding (N-1)-tuplet loss can be formulated as follows:

$$L_{\text{Npair}} = \frac{1}{Q}\sum_{i=1}^{Q}\left\{\log\left[1 + \sum_{i\neq j}\exp\left(\psi_i^T\psi_j' - \psi_i^T\psi_i'\right)\right]\right\}, \tag{6}$$

where $\psi_i = f(p_i)$ and $\{p_i, p_1', p_2', \cdots, p_Q'\}$ denote Q pairs of examples from Q different classes. Here, p_i and p_i' indicate the query and the positive sample, respectively. $\{p_j', j \neq i\}$ is defined as the negative samples.

Both the triplet loss and the N-pair loss use distance as a similarity measure, which is sensitive to changes in scale. The angular loss minimizes the angle at ψ_n of a triangle formed from the three embeddings, instead of simply minimizing the distance of ψ_p to ψ_a relative to ψ_n. Angular loss is both rotationally invariant and scale invariant, using a triangle in which all edges of the triplet are taken into account [33]. However, under certain circumstances, the minimization of the angle at $\angle n$ will push ψ_n toward ψ_a as exhibited in Fig. 5 by the two gray triangles and the dashed arrow. A new triangle ΔMNK is formulated to move the anchor sample ψ_a and positive sample ψ_p to ψ_m and ψ_k separately (see the red triangle in Fig. 5). The edges are denoted as ε_{mn}, ε_{nk}, and ε_{mk}.

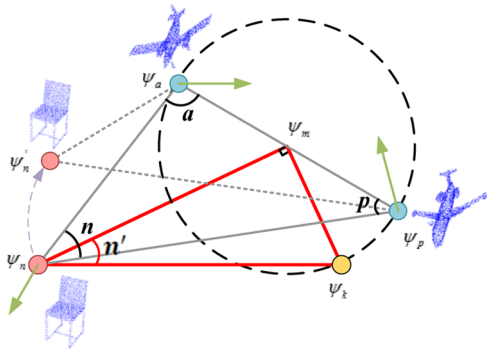

Fig. 5. Illustration of angular loss.

First, a hypersphere $\mathcal{G}_C$ passing through ψ_a and ψ_p, with a midpoint ψ_m, is constructed, where $\psi_m = (\psi_a + \psi_p)/2$. Second, a straight line ε_{nk} is found by passing through the midpoint ψ_m, setting an auxiliary line perpendicular to the line ε_{mn} and intersecting the hypersphere Θ_C at the point ψ_k. Finally, a tangent function of angular loss is constructed in the right-angle triangle by ψ_m, and ψ_k is minimized as follows:

$$\tan\angle n' = \frac{\|\psi_m - \psi_k\|}{\|\psi_m - \psi_n\|} = \frac{\|\psi_a - \psi_p\|/2}{\|\psi_m - \psi_n\|} \le \tan\delta, \tag{7}$$

where the hyperparameter δ is predefined upper accepted bounds of the loss, which is set to be between 30° and 50° in our experiments.

Equation (7) can be rewritten as Eq. (8):

$$S(\psi_a, \psi_p) \le 4\tan^2\delta\, S(\psi_m, \psi_n). \tag{8}$$

The angular loss function is defined as follows:

$$L_{\text{ang}} = \max(0,\ S(\psi_a, \psi_p) - 4\tan^2\delta\, S(\psi_m, \psi_n)). \tag{9}$$

Since our work is based on unsupervised learning and learns from the idea of instance discrimination in [36], in this paper, the anchor point indicates the embedded local features, the positive sample indicates the global features of one object, and the negative samples indicate the global features of other objects. However, there are different channels between the local features $\{F_{pi}^{(l)}, \forall i, l\}$ and global features G. MLPs are used to embed them into a shared feature space, e.g., $\Gamma^{(l)}(F_{pi}^{(l)}), \mathrm{H}(G)$.

For each sample from a mini-batch of size Q, $\{G_i\}_{j=1}^{q}$ denotes the global features of different objects. Therefore, Eq. (6) can be rewritten as follows:

$$L_N = \frac{1}{Q}\sum_{i,l}\left\{\log\left[1 + \sum_{G_j\neq G}\exp\left(\Gamma^{(l)}\left(F_{pi}^{(l)}\right)^T\mathrm{H}(G_j) - \Gamma^{(l)}\left(F_{pi}^{(l)}\right)^T\mathrm{H}(G)\right)\right]\right\}. \tag{10}$$

The angular loss, which is combined in a *log-sum-exp* formulation, can be formed as Eq. (11):

$$L_A = \frac{1}{Q}\sum_{i,l}\left\{\log\left[1 + \sum_{G_j\neq G}\exp(e)\right]\right\}, \tag{11}$$

$$e = 4\tan^2\delta\ \rho \exp\left(\Gamma^{(l)}\left(F_{pi}{}^{(l)}\right) + \mathrm{H}(G_j)\right)^T \mathrm{H}(G_j)$$
$$- 2\left(1 + 4\tan^2\delta\right)\rho \exp\left(\Gamma^{(l)}\left(F_{pi}{}^{(l)}\right)^T \mathrm{H}(G_j)\right), \quad (12)$$

where ρ is set to a constant value of 64 to rescale the features.

The angular constraint can be combined with traditional distance metric loss to boost the overall performance. The improved angular loss is defined as follows:

$$L_{\mathrm{IA}} = \tau L_A + L_N, \quad (13)$$

where τ denotes a weight, which is set to 0.1 in all experiments.

2. Self-Reconstruction Loss

Chamfer distance (CD) is chosen as the reconstruction error.

$$L_R = \frac{1}{|P|}\sum_{p\in P}\min_{p'\in D(G)}\left\|p - p'\right\|_2^2 + \frac{1}{|D(G)|}\sum_{p'\in D(G)}\min_{p\in P}\left\|p' - p\right\|_2^2. \quad (14)$$

The average nearest squared distance between the reconstructed point cloud $D(G)$ and the input point cloud P is measured by CD in Eq. (14). The point cloud $D(G)$ is generated by our proposed decoder conditioned on the global features G.

3. Total Loss

Accordingly, the total loss of the training network involves a combination of the above losses, which is defined in Eq. (15):

$$L_{\mathrm{com}} = L_R + L_{\mathrm{IA}}. \quad (15)$$

3. EXPERIMENTS AND RESULTS

A. Datasets and Implementation Details

Datasets. To demonstrate the effectiveness and efficiency of our network, extensive experiments are conducted on ModelNet40 and ModelNet10 benchmarks and the real-world dataset (the 3D Terracotta Warriors fragments) for the point cloud classification task. ModelNet10 contains 4899 CAD models from 10 categories and is split into 3991 for training and 908 for testing. ModelNet40 comprises 9843 training objects and 2468 test objects in 40 classes. 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.

The raw point clouds of the Terracotta Warrior fragments are scanned by trained students in our lab using Creaform VIU 718 handheld 3D scanner from Canada. Figure 6 illustrates some obtained models of Terracotta Warriors fragments. There are conflicts between the dense point cloud obtained by the high resolution of 3D scanners and the number of the neural network input point clouds. To this end, the sampled point clouds are obtained using Geomagic software as exhibited in Fig. 7(a). According to the parts of the Terracotta Warriors, data can be divided into four categories: Arm (800), Body (810), Head (810), and Leg (830). The number of Terracotta Warrior fragments is not far from enough for training deep neural networks, and the deep neural networks have a fixed input number as 1024 or 2048. Therefore, an extended dataset is generated by random

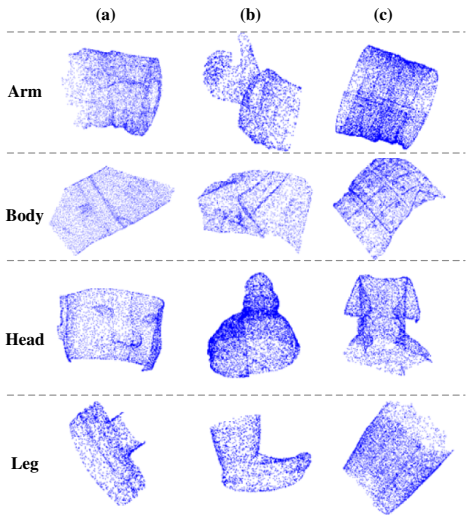

Fig. 6. Illustration of the Terracotta Warriors fragments.

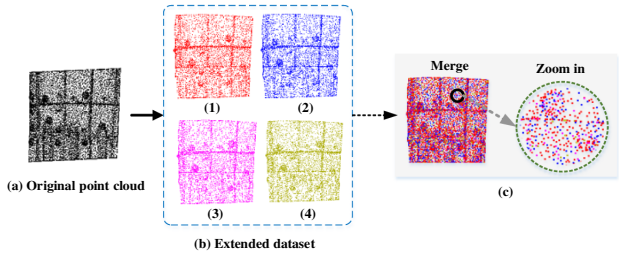

Fig. 7. Pipeline of input data generation approach.

Table 1. Number of Fragments for Each Class in the Expanded Dataset

Label	Arm	Body	Head	Leg	Total
Train	2656	2720	2720	2496	10144
Test	476	504	428	444	1852

sampling from the dense point clouds [Fig. 7(b)]. Here we ensure that the four non-overlapping point clouds are sampled from the same object by the random sample method, which is exhibited in Fig. 7(c). Subsequently, the extended dataset that included 11,996 patches of the 3D Terracotta Warriors fragments are obtained, and the number of samples included in the different partitions is exhibited in Table 1.

Network configuration. In UMA-Net, there are three multi-scale shell blocks. In each block_ i, the parameters, M_i, X_i, N^i_{shell}, and D_i $(i = 1, 2, 3)$, respectively denote the number of sampled points by FPS, the number of shells, the number of points in each shell of the same scale, and the output channels. M_i is set to [512,128,32], X_i is set to [4,2,1], N^i_{shell} is set to [8,16,32], and D_i is set to [256,89,1792] Therefore, points in each area of the local region can be defined as $X_i \times N^i_{\mathrm{shell}}$. To measure the similarity between local features in each encoding layer and the global feature, the MLPs are used to embed them into a shared feature space. Here, the dim is set to 512.

Training. The network was trained for 200 epochs on an NVIDIA GTX 1080Ti GPU and PyTorch v1.2, using an Adam optimizer without weight decay. An initial learning rate of 0.01,

an initial momentum for batch normalization layers of 0.9, and a batch size of 22 were set. The learning rate was decayed by 0.7 every 20 epochs using the lambda learning rate scheduler, and the momentum was decayed by 0.5 every 20 epochs. The batch-normalization and Rectified Linear Unit (ReLU) activation were applied to each layer, and dropout was used with $p = 0.5$.

B. Experiments on Modelnet40

1. Pre-training the Network

UMA-Net is trained to receive and reconstruct point clouds of 1024 points. During training, a random translation in [0.2, 0.2] and a random anisotropic scaling in [0.67, 1.5] are applied to augment the input data. The model is pre-trained across all categories of the Modelnet40 dataset, and then the trained model is transferred to the classification downstream task. All pre-trained weights are frozen during training, and the network is not fine-tuned for the downstream task.

2. Classification

To evaluate the performance of the model on representation learning, we extract the global features G of ModelNet40/10 from the pre-trained model without any fine-tuning, train a linear classifier on them, and report the classification accuracy. The performance comparison between our UMA-Net with several baselines is summarized in Table 2. For unsupervised representation learning methods, we also compare with some advanced voxel- [37] and view-based [38] methods except for point-based methods. From the results of the experiment conducted on Modelnet10 in Table 2, our unsupervised learning method outperformed all unsupervised methods and is slightly inferior to the supervised methods. For the results of ModelNet40, our model even outperforms the methods trained under the larger ShapeNet dataset, such as FoldingNet [10] and l-GAN [16]. For a fair comparison, we train all these methods under ModelNet40. Our model reaches a much better classification performance with an accuracy of 92.06%, outperforming the approaches FoldingNet, l-GAN, Multi-Task, and L2G-AE by 7.7%, 4.79%, 2.96%, and 1.42%, respectively. Results suggest that our model achieves optimal accuracy on the ModelNet40 classification task compared to other unsupervised methods listed in Table 2. Our model is still 1.16% better than SO-Net using 2048 points. In general, the results also suggest that the unsupervised model is competitive with the supervised models.

Table 2. Comparisons of the Classification Accuracy (%) of Our Method Against the Unsupervised and Supervised Methods on ModelNet40/10[a]

Method	Re.	Supervised	M40	M10
3D-GAN [37]	V	F	83.30	91.00
VIPGAN [38]	Mv	F	91.98	94.05
FoldingNet (M40) [10]	P	F	84.36	91.85
FoldingNet [10]	P	F	88.40	94.40
l-GAN (M40) [16]	P	F	87.27	92.18
l-GAN [16]	P	F	85.70	95.30
Multi-Task [22]	P	F	89.10	—
L2G-AE [17]	P	F	90.64	95.37
PointNet [4]	P	T	89.20	—
PointNet++ [5]	P	T	90.70	—
ShellNet [9]	P	T	93.10	—
PointWeb [8]	P	T	92.30	—
AMS-Net [29]	P	T	92.94	95.83
DGCNN [7]	P	T	92.90	—
SO-Net [15]	P(2048)	T	90.90	—
AMS-Net [29]	P, N	T	93.52	95.91
Ours	P	F	92.06	95.60

[a]M40, ModelNet40; M10, ModelNet10; Re., the representation of input; Acc., overall accuracy; V, voxel; Mv, multi-views; P, xyz coordinates of the point; N, surface normal vector.

3. Visualization of the Global Features G

To reflect the ability of the encoder by visualizing the representative features, we select 10 categories and randomly sample 200 shapes from each category. The t-SNE visualization is exhibited in Fig. 8, which demonstrates the performance of our UMA-Net.

4. Robustness

The robustness of our UMA-Net on sampling density was evaluated by using sparser points of numbers 1024, 512, 256, and 128 as the input to a model trained with 1024 points. Figure 9 suggested that our UMA-Net drops by 1.58% with points of numbers from 1024 to 512 and 4.89% with points of numbers from 1024 to 256. When the number of sampling points is set as 128, the UMA-Net we proposed can still reach 82.33%. Besides, compared with PointNet and SO-Net, our method has a gentler decline and is still considerably robust.

5. Ablation Study

In this section, an extensive ablation study was performed to investigate the effectiveness of each component of the UMA-Net architecture as follows. (1) The structure of the encoder: single-scale versus multi-scale. As suggested in Table 3, the UMA-Net outperforms the single-scale model by 1.42%, which is owing to our network being able to extract multi-scale detail features effectively. (2) The structure of decoder: MLPs versus FoldingNet versus Cross-Dim block. The folding-based decoder has the same accuracy as our model. To further prove

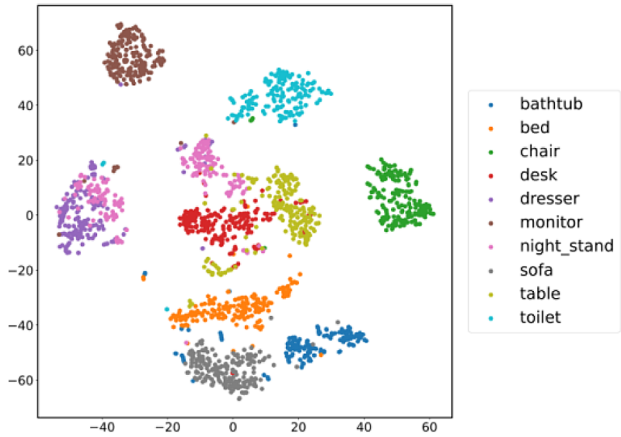

Fig. 8. Visualization of embedded features with T-SNE.

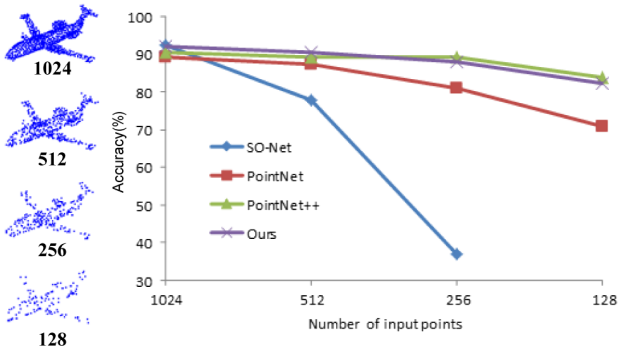

Fig. 9. Robustness for sparser point clouds.

Table 3. Ablation Study of Our Method

		Acc. (%)
Encoder	Single-scale	90.64
	Multi-scale	92.06
Decoder	MLPs	91.65
	FlodingNet	92.06
	Cross-dim block	91.97
Loss	L_R	89.47
	$L_R + L_N$	91.86
	$L_R + L_A$	89.59
	$L_R + L_{\mathrm{IA}}$	92.06
N^i_{shell}	[4,8,16]	92.00
	[8,16,32]	92.06
	[16,32,32]	90.88

the effectiveness of our model, the convergence of our UMA-Net and the comparison method are presented as exhibited in Fig. 10. In the first 50 iterations of training, our model has stronger convergence than the other two. In the subsequent 100 iterations, the three networks gradually become flat and stable. In conclusion, our UMA-Net can generate a result that is more similar to the input point cloud. (3) Loss: the notation of each loss function has been defined in Section 3.D. As suggested in Table 3, the result of the total loss ($L_R + L_{\mathrm{IA}}$) achieves the best performance. The angular loss contributes to the network. In summary, the integration of multi-scale encoder, cross-dim decoder, and angular loss achieve significant performance improvement over baseline. (4) The number of the neighbors in each scale is $N_{\mathrm{nei}} = X \times N_{\mathrm{shell}}$. Here, we also perform the fixed value of the number of shells X in [9]. The shell size N_{shell} is the key factor in affecting the neighbors' regions. The performance of our UMA-Net is tested with different shell sizes. In each block$_i$, we set N^i_{shell} to [8,16,32] for point cloud classification.

6. Complexity Analysis

We measure network complexity by floating point operations (FLOPs) and GPU inference throughput on NVIDIA GTX 1080Ti GPU. With batch size 22, point cloud size 1024 from the ModelNet40 dataset. Table 4 summarizes the model complexity of different methods. We can see our large model requires considerable computation cost but maintains an acceptable

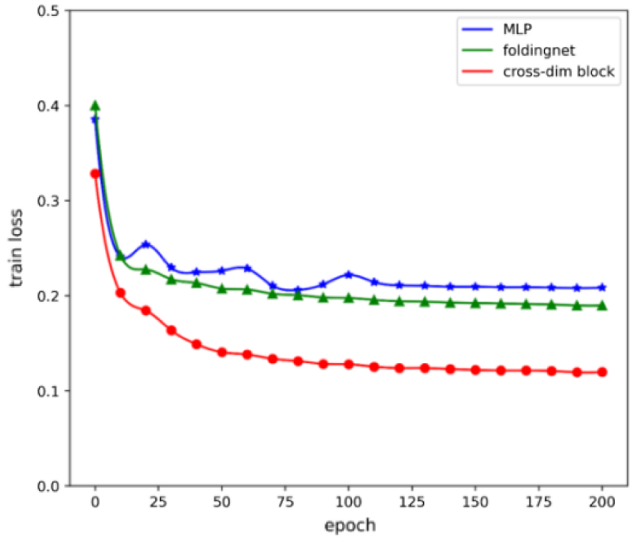

Fig. 10. Contrast of convergence.

Table 4. FLOPs and Inference Throughput of Several Models[a]

Model	FLOPs	Throughput	Acc. (%)
DGCNN	39.924 G	257.27pc/s	92.9
SSG PointNet++	13.814 G	113.42pc/s	90.5
MSG PointNet++	64.473 G	68.78pc/s	91.7
Method in [28]	91.666 G	457.59pc/s	—
Ours	8.039 G	53.84pc/s	92.06

[a]pc/s, point cloud(s) per second.

actual cost on GPU due to the simplicity of the SSG model. Our UMA-Net achieves a better trade-off on speed and accuracy.

C. Application on the Terracotta Warrior Dataset

To prove the effectiveness of our UMA-Net on 3D Terracotta Warrior fragments, the performance comparison between our model with several supervised baselines is exhibited in Table 5. The first two are traditional methods, and the others are DML ones. The highest mean accuracy of the existing traditional approaches is 87.64%. Our UMA-Net can achieve competitive performance with an accuracy improvement of 6.26% compared to the method in [39]. Simultaneously, our AMS-Net improves accuracy by 4.97% compared to PointNet. Although the method in [28] is a dual modal that incorporates geospatial and texture information of the fragments, our method outperforms it by 2.49%. However, our UMA-Net is only 1.78% lower than the supervised learning method named AMS-Net. Compared with ModelNet40, the real-world dataset has the following characteristics: the smooth surface of objects and uneven distribution of wrinkles, such as arms and bodies. To sum up, the results suggest that our unsupervised model is competitive with the supervised models, and these quantitative results further validate the high effectiveness and promises of UMA-Net.

Table 6 exhibits the accuracy of the average class. From the results, first, a conclusion can be drawn that the accuracy of the class Body is the highest, while the accuracy of the class Arm is the lowest. The main reason is that most of the body parts are

Table 5. Compared with Other Methods on the 3D Terracotta Warrior Fragment Dataset[a]

Method	Data Type	Supervised	OA(%)
Method in [26]	I	T	84.34
Method in [39]	P	T	87.64
PointNet [4]	P	T	88.93
Method in [28]	P, I	T	91.41
AMS-Net [29]	P	T	95.68
Ours	P	F	93.90

[a]I, the image.

Table 6. Classification Accuracies of the Four Classes[a]

Method	Arm	Body	Head	Leg
Method in [28] (G)	82.51	96.45	92.36	84.41
Method in [28] (T)	77.75	92.75	91.50	76.25
Method in [28]	87.55	87.55	94.37	88.41
AMS-Net [29]	92.40	98.10	98.00	94.20
Ours	91.00	94.73	94.58	92.54

[a]G, the result of the classification based on geospatial information only; T, texture information.

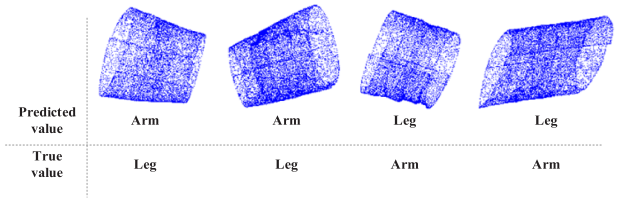

Fig. 11. Illustration of failure examples.

Table 7. Results for Different Numbers of Input Points

Input Numbers	1024	512	256	128
Acc. (%)	93.90	93.51	92.29	90.43

wearing armor, or the clothes have more folds, as illustrated in Fig. 6. Second, the characteristics of the class Body are more obvious in general. Finally, the characteristics of class Arm are similar to class Leg, e.g., the models of column *c* in these two classes as illustrated in Fig. 6. The fragments in the two columns on the left of Fig. 11 belong to class Arm, and the remaining two columns on the right are from class Leg. As the features of class Leg are relatively smooth, these two categories can easily be misclassified.

From Table 7, we know that the robustness of our UMA-Net on sampling density is equally applicable to the real-world dataset.

4. CONCLUSION

In this paper, a novel autoencoder network for unsupervised representation learning has been proposed. The main contributions can be summarized as follows: (1) Our model can capture the structural and semantic information at the same time by the similarity measure between local features and global features. (2) A hierarchical multi-scale shell-based encoder is present to learn the correlation between different areas for point clouds. (3) A structure-preserving decoder is proposed for high-quality point cloud generation in a coarse-to-fine manner. (4) It is the first attempt to our knowledge to apply the unsupervised representation learning for 3D Terracotta Warrior fragments. Extensive experiments conducted on challenging benchmarks have demonstrated that the model we proposed achieves excellent performance on 3D object classification, significantly narrowing the gap with the fully supervised counterparts in the literature. In the future, we would like to exploit unsupervised techniques to improve the performance and extend our approach to high-level 3D understanding tasks such as 3D object detection and semantic segmentation. Additionally, we hope the findings in this study will encourage more research on 3D representation learning.

Funding. National Key Research and Development Program of China (2019YFC1521103, 2019YFC1521102, 2020YFC1523301); National Natural Science Foundation of China (61731015, 61701403); Key R&D Projects in Shaanxi Province (2019ZDLSF07-02, 2019ZDLGY10-01); Key R&D Projects in Qinghai Province (2020-SF-140); China Postdoctoral Science Foundation (2018M643719); Young Talent Support Program of the Shaanxi Association for Science and Technology (20190107).

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

Disclosures. The authors declare no conflicts of interest.

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

[†]These authors contributed equally to this paper.

REFERENCES

1. C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in *IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 918–927.
2. L. Ge, Y. Cai, J. Weng, and J. Yuan, "Hand PointNet: 3D hand pose estimation using point sets," in *IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 8417–8426.
3. K. Zhang, C. Xiong, W. Zhang, H. Liu, D. Lai, Y. Rong, and C. Fu, "Environmental features recognition for lower limb prostheses toward predictive walking," IEEE Trans. Neural Syst. Rehabil. Eng. **27**, 465–476 (2019).
4. C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: deep learning on point sets for 3D classification and segmentation," in *International Conference on Computer Vision and Pattern Recogintion(CVPR)*, Honolulu, Hawaii, USA, 2017, pp. 652–660.
5. C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: deep hierarchical feature learning on point sets in a metric space," in *31st Annual Conference on Neural Information Processing Systems (NIPS)*, Long Beach, California, USA, 2017, pp. 5099–5108.
6. C. Wang, B. Samari, and K. Siddiqi, "Local spectral graph convolution for point set feature learning," in *European Conference on Computer Vision (ECCV)*, Munich, Germany, 2018, pp. 52–66.
7. Y. Wang, Y. B. Sun, Z. W. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," ACM Trans. Graph. **38**, 146 (2019).
8. H. S. Zhao, L. Jiang, C. W. Fu, and J. Y. Jia, "PointWeb: enhancing local neighborhood features for point cloud processing," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, California, USA, 2019, pp. 5550–5558.
9. Z. Zhang, B.-S. Hua, and S.-K. Yeung, "ShellNet: efficient point cloud convolutional neural networks using concentric shells statistics,"

in *IEEE/CVF International Conference on Computer Vision* (2019), pp. 1607–1616.
10. Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: point cloud auto-encoder via deep grid deformation," in *IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 206–215.
11. D. Li, W. Hung, J. Huang, S. Wang, N. Ahuja, and M. Yang, "Unsupervised visual representation learning by graph-based consistent constraints," in *European Conference on Computer Vision* (2016), pp. 678–694.
12. S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.
13. D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: feature learning by inpainting," in *IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 2536–2544.
14. M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *European Conference on Computer Vision* (Springer, 2016), pp. 69–84.
15. J. Li, B. M. Chen, and G. H. Lee, "SO-Net: self-organizing network for point cloud analysis," in *International Conference on Computer Vision and Pattern Recogintion (CVPR)*, Salt Lake City, Utah, USA, 2018, pp. 9397–9940.
16. P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *International Conference on Machine Learning (PMLR)* (2018), pp. 40–49.
17. X. Liu, Z. Han, X. Wen, Y.-S. Liu, and M. Zwicker, "L2G auto-encoder: understanding point clouds by local-to-global reconstruction with hierarchical self-attention," in *27th ACM International Conference on Multimedia* (2019), pp. 989–997.
18. M. Gadelha, R. Wang, and S. Maji, "Multiresolution tree networks for 3D point cloud processing," in *European Conference on Computer Vision (ECCV)* (2018), pp. 103–118.
19. C.-L. Li, M. Zaheer, Y. Zhang, B. Poczos, and R. Salakhutdinov, "Point cloud GAN," in *6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.
20. Y. Sun, Y. Wang, Z. Liu, J. Siegel, and S. Sarma, "PointGrow: autoregressively learned point cloud generation with self-attention," in *IEEE/CVF Winter Conference on Applications of Computer Vision* (2020), pp. 61–70.
21. F. Poux, C. Mattes, and L. Kobbelt, "Unsupervised segmentation of indoor 3D point cloud: application to object-based classification," Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. **XLIV-4/W1-2020**, 111–118 (2020).
22. K. Hassani and M. Haley, "Unsupervised multi-task feature learning on point clouds," in *IEEE/CVF International Conference on Computer Vision* (2019), pp. 8160–8171.
23. L. Zhang and Z. Zhu, "Unsupervised feature learning for point cloud understanding by contrasting and clustering using graph convolutional neural networks," in *International Conference on 3D Vision (3DV)* (IEEE, 2019), pp. 395–404.
24. N. A. Rasheed and M. J. Nordin, "Using both HSV color and texture features to classify archaeological fragments," Res. J. Appl. Sci. Eng. Technol. **10**, 1396–1403 (2015).
25. L. Y. Qi and K. G. Wang, "Kernel fuzzy clustering based classification of ancient-ceramic fragments," in *International Conference on Information Management and Engineering*, Chengdu, China, 2010, pp. 348–350.
26. Z. Lu, C. Li, G. Geng, P. Zhou, Y. Li, and Y. Liu, "Classification of cultural fragments based on adaptive weights of multi-feature descriptions," Laser Optoelectron. Prog. **57**, 321–329 (2020).
27. G. H. Geng, J. Liu, X. Cao, Y. Y. Liu, and M. Q. Zhou, "Simplification method for 3D terracotta warrior fragments based on local structure and deep neural networks," J. Opt. Soc. Am. A **37**, 1711–1720 (2020).
28. K. Yang, X. Cao, G. H. Geng, K. Li, and M. Q. Zhou, "Classification of 3D terracotta warriors fragments based on geospatial and texture information," J. Visualization **24**(2), 251–259 (2021).
29. 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**, 3713 (2021).
30. S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2005), pp. 539–546.
31. E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition* (Springer, 2015), pp. 84–92.
32. F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: a unified embedding for face recognition and clustering," in *IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 815–823.
33. J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," in *IEEE International Conference on Computer Vision* (2017), pp. 2593–2601.
34. H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4004–4012.
35. K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in Neural Information Processing Systems* (2016), pp. 1857–1865.
36. Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 3733–3742.
37. J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," in *30th International Conference on Neural Information Processing Systems* (2016), pp. 82–90.
38. Z. Han, M. Shang, Y.-S. Liu, and M. Zwicker, "View inter-prediction GAN: unsupervised representation learning for 3D shapes by learning global shape memories to support local view predictions," in *AAAI Conference on Artificial Intelligence* (2019), pp. 8376–8384.
39. G. Q. Du, M. Q. Zhou, C. L. Yin, Z. K. Wu, and W. Y. Shui, "Classifying fragments of terracotta warriors using template-based partial matching," Multimedia Tools Appl. **77**, 19171–19191 (2018).