X-3D: Explicit 3D Structure Modeling for Point Cloud Recognition

Supplementary Material

A. Supplementary Material on Sec 5.6

A.1. Manifold Learning

There are two views of manifold learning, ISOMAP optimizes by minimizing the distance vector between the embedding space and the input space, and its procedure can be written as:

$$\underset{B}{argmin} \sum_{i,j} ||dist_A(A_i, A_j), dist_B(B_i, B_j)|| \quad (1)$$

where $dist_A$ usually stands for a distance metric such as geodesic distance that conforms to the properties of the underlying manifold in original space A, while $dist_B$ usually refers to Euclidean distance in the embedding space B.

Then, LLE works by minimizing the local structure of the embedding space and the input space:

$$ES = \underset{ES}{argmin} \sum_{i=1}^{N} ||A_i - \sum_{j=1}^{k} ES_{i,j} A_{i,j}|| \quad (2)$$

Then the embedding space B is obtained by satisfying the linear reconstruction relation, which means that the LLE generates a mapping from the original input to embedding through the geometric local structure:

$$B = \underset{B}{argmin} \sum_{i=1}^{N} ||B_i - \sum_{j=1}^{K} ES_{i,j} B_{i,j}|| \quad (3)$$

By the comparison of (1) and (4), (2) and (7), it can be seen that whether explicit local structure is used in the process of embedding relation vectors is the main difference between the two methods, so LLE provides some interpretability for the efficient performance of X-3D.

A.2. Manifold in Point Clouds.

Point cloud is typical manifold data, so many works [16, 20, 28, 39, 47, 57]directly apply manifold learning to point cloud tasks. GP-PCS [28] employs Gaussian processes suitable for functions defined on Riemannian manifolds to model the surface on the point cloud. DeepUME [20] directly uses manifold embedding to complete point cloud registration. PointManifold [47] decomposes the 3D point cloud neighborhood into three 2D manifolds and extracts features separately. Some works [16, 39] believe that it is more important to use distance embeddings in manifold space such as geodesic distance instead of Euclidean embeddings.

Inspired by these works, we argue that existing works may learn a mapping from Euclidean embeddings to manifold embeddings by stacking a large number of MLPS, which is similar to manifold learning. In Figure 1, we compare the similarity between geometric correlations captured by existing ISHM models and geodesic distances, which further proves the association of IHSM with ISOMAP.

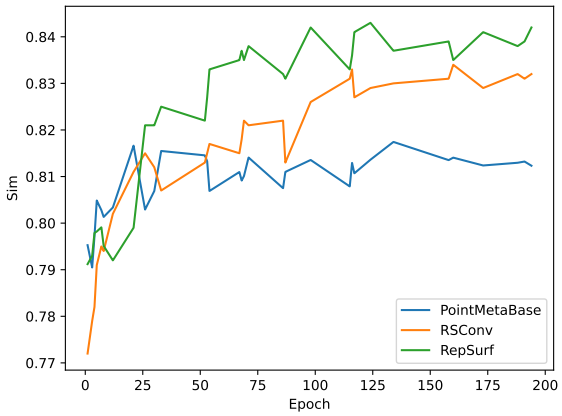


Figure 1. We compare the similarity between geometric correlations captured by existing ISHM models and geodesic distances, and can find that existing works implicitly learn a mapping from Euclidean embeddings to manifold embeddings and capture local structure through manifold embeddings.).

However, from the Figure 1, we can see that even though the learning difficulty can be reduced by adding additional surface information, such as RepSurf curvature, etc., due to the lack of explicit local structure, it is still very difficult to directly learn the Euclidean embedding to the manifold embedding (the similarity is only about 84%).

B. Examples of Implict High-dimensional Structure Modeling

In this section, we will detail some representative works on implicit high-dimensional structure modeling. There are mainly two key parts, one is how to construct the relation vector, and the other is to generate the dynamic embedding kernel for the relation vector.

PointNet++ [30] and **PointMetaBase** [22] simply treat the relative coordinate difference as a relation vector:

$$V_{i,j} = \mathbf{p}_i - \mathbf{p}_{i,j} \quad (4)$$

RandLA [15] additionally introduces absolute coordi-

nates and Euclidean distances:

$$V_{i,j} = (\mathbf{p}_i - \mathbf{p}_{i,j}, \mathbf{p}_i, \mathbf{p}_{i,j}, ||\mathbf{p}_i - \mathbf{p}_{i,j}||_2^2) \tag{5}$$

GAM [14] proposes a simplified depth gradient information as the relation vector:

$$\mathbf{p}_i - \mathbf{p}_{i,j} = (\vec{x_{i,j}}, \vec{y_{i,j}}, \vec{z_{i,j}}) \tag{6}$$

$$V_{i,j} = (\frac{\vec{z_{i,j}}}{||\mathbf{p}_i - \mathbf{p}_{i,j}||_2^2} \frac{\vec{x_{i,j}} + \vec{y_{i,j}}}{\sqrt{(\vec{x_{i,j}})^2 + (\vec{y_{i,j}})^2}}, ||\mathbf{p}_i - \mathbf{p}_{i,j}||_2^2) \tag{7}$$

RepSurf [34] is special. Although it captures surface information at the beginning, as the down-sampling rate becomes larger, the point cloud becomes sparse, and the initial surface information can only describe points, but cannot characterize the neighborhood structure.

$$V_{i,j} = [\mathbf{p}_i - \mathbf{p}_{i,j}, n_j, n_j^T \mathbf{p}_{i,j}, S_{i,j}] \tag{8}$$

where n_j represents the normal vector of the j-th point and $S_{i,j}$ means the coordinate of the j-th point in the spherical coordinate system with point i as the center.

Then for vector kernel, most methods like PointNet++ [30], PointNeXt [33] and so on directly use shared-weight MLP to implicitly embed the relation vector:

$$\mathcal{M}(\mathbf{W}_{i,j}, \mathbf{f}_{i,j}, V_{i,j}) = MLP(V_{i,j}, f_{i,j}) \tag{9}$$

At the same time, there are also many methods such as PointMetaBase [22], RepSurf [34], etc. that explicitly use shared-weight MLP to embed relationship vectors:

$$\mathcal{M}(\mathbf{W}_{i,j}, \mathbf{f}_{i,j}, V_{i,j}) = MLP(V_{i,j}) + MLP(f_{i,j}) \tag{10}$$

RSConv [23] uses MLP to generate dynamic kernels based on relation vectors:

$$\mathbf{W}_{i,j} = MLP(V_{i,j}) \tag{11}$$

$$\mathcal{M}(\mathbf{W}_{i,j}, \mathbf{f}_{i,j}, V_{i,j}) = MLP(\mathbf{W}_{i,j} * \mathbf{f}_{i,j}) \tag{12}$$

KPConv [41] dynamically generates kernels based on geometric relationships between neighborhood points and predefined kenrel points:

$$\mathbf{W}_{i,j} = \sum_{k=1}^{r} h(\mathbf{p}_{i,j}, \widetilde{\mathbf{p}}_k)\mathbf{G}_k \tag{13}$$

$$\mathcal{M}(\mathbf{W}_{i,j}, \mathbf{f}_{i,j}, V_{i,j}) = \mathbf{W}_{i,j}\mathbf{f}_{i,j} \tag{14}$$

where $\widetilde{\mathbf{p}}_k$ and $\mathbf{G}_k$ represent the coordinates and weights of the k-th predefined kernel points, and r means the number of kernel points. h represents the geometric distance between kernel point and neighborhood point.

It can be seen that these methods implicitly capture the local structure in the high-dimensional space through the relation vector, so they pay more attention to how to construct a good relation vector $V_{i,j}$ and how to generate a good vector kernel $\mathbf{W}_{i,j}$ for the relation vector

Algorithm 1 Pytorch-Style Pseudocode of PointHop

```
# b: batch size, n: number of center points
#k: number of neighbor points, c: feature channels
#p: coordinates of input point cloud (b,n,k,c)
partion_idx=(p[...,0]>0)*4+(p[...,1]>0)*2+(p
    [...,2]>0)
centroid=scatter_mean(p,idx=partion_idx)
centroid=concat(centroid,dim=-1)
return centroid
```

Algorithm 2 Pytorch-Style Pseudocode of Neighborhood Context Propogation

```
# b: batch size, n: number of center points
#k: number of neighbor points, c: feature channels
#f: input point cloud
#fps_idx: index of the center points
group_idx=ball query(f) #b,n,k
f=grouping&modeling(f,group_idx) #b,n,k,c
f_0=Maxpooling(f) #b,n,c

f_1=scatter_mean(f.view(b,-1,c),dim=1,index=
    group_idx.view(b,-1))
f_1=gather(f_1,fps_idx)

f_2=MLP([f_0,f_1])
return f_2
```

C. Implementation of X-3D

In this part, we detail a part of the implementation of X-3D.

First, we introduce the implementation algorithm of PointHop which is shown in Algorithm 1, we use the scatter operator to compute the centroid of each partition.

Then, We detail the specific implementation of neighborhood context propagation which is shown in Algorithm 2. We use the scatter operator to fuse the features of each point in different neighborhoods (either as a center point or as a neighboring point), and then select only the center point's fused feature as the propagated feature.

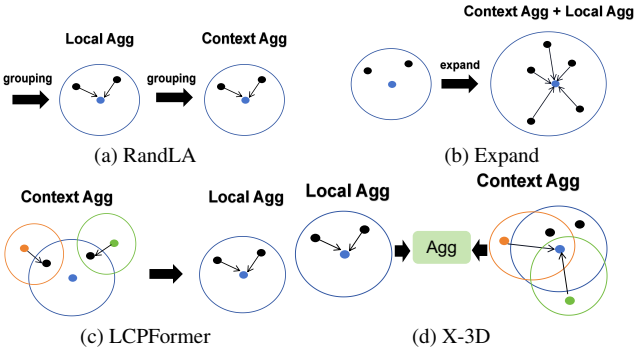
D. Additional Analysis of NCP

In this part, we conduct detailed analysis and ablation experiments for Neighborhood Context Propagation.

Firstly, from the perspective of computation, NCP simply uses the overlapping region to propagate context information and expand the receptive field, which saves a lot of computation compared with simply expanding the neighborhood range [18].

Then we compared methods for different propagation contexts which is shown in Figure 2. After calculating

the features of each point through the attention mechanism, LCPFormer first propagates the context information through context agg, and then performs local agg to aggregate the features in the neighborhood. RandLA first performs local agg to extract local features after obtaining the neighborhood by grouping, and then performs another grouping to carry out context agg to propagate context information with the local features obtained in the previous step. In X-3D, we believe that local agg captures fine local details, while context agg captures coarse global information. In order to balance the role of the two steps, we discard the sequential execution of the two steps, but perform them in parallel, and fuse the two information through an adaptive aggregation module.



(a) RandLA (b) Expand (c) LCPFormer (d) X-3D

Figure 2. We visualize how different methods propagate the neighborhood context. We see that previous methods usually execute context agg and local agg in a different order or directly expand the neighborhood to execute both. X-3D executes the two steps in parallel, which not only preserves the fine local details of the current region but also fuses the context information

Furthermore, we see that we only propagate the context information including the center point, which effectively limits the scope of propagation and avoids conflicts caused by large differences between local structures in different regions.

Finally, to demonstrate the effectiveness of the above design, we conduct ablation experiments of NCP on S3DIS Area5. We see that the parallel local agg and context agg provide the model with the structure to deal with context flexibly. At the same time, by limiting the propagation range, the conflict between local structures is avoided, which effectively improves the performance of NCP

E. Additional Analysis of ES

In this section, we first introduce several explicit local structures in depth and detail their properties. Then we analyze the reasons for the excellent performance of X-3D in detail for the viewpoints of ES properties.

Table 1. We tested NCP on S3DIS, where L→C stands for local agg followed by context agg, L& C stands for performing both steps in parallel, and Includ-Center stands for propagating only context information containing the center point

L→ C	C→ L	L& C	Include-Center	mIoU
		✓	✓	**71.9**
✓			✓	71.2
	✓		✓	70.9
		✓		71.6

E.1. Properties

First we present the implementation of different explicit local structures:

LR: LLE [36] proposes local linear relations (LR) as explicit local structures which is shown in (2),

PCA: [6, 19] proposed to obtain an explicit local structural descriptor by performing pca on the local neighborhood and combining the eigenvalues and eigenvectors. Specifically, for eigenvectors (e_1,e_2,e_3) and eigenvalues (λ_1,λ_2,λ_3), we can describe the overall shape by linear, planar, and scatter:

$$linear = \frac{\lambda_1 - \lambda_2}{\lambda_1}, \quad planar = \frac{\lambda_2 - \lambda_3}{\lambda_1}, \quad scatter = \frac{\lambda_3}{\lambda_1} \tag{15}$$

PH: PointHop [51] partitions the point cloud neighborhood into fixed octets and computes the centroid of each partition. The process of PH we have described in detail in (8).

Then, we then detail the different properties:

Symmetry: Since the point cloud is unordered, an effective ES should possess symmetry and not be affected by the order of the point cloud. According to the above formula, we see that the linear relation ES defined by LLE is obviously affected by the order of the points, so its property of symmetry is missing.

Gloabl Shape: The global shape of a local region describes the overall shape information, such as sphere, cuboid, etc., which provides a good geometric prior for the region category. Traditional point cloud analysis methods can usually identify the object shape information by the main direction of the data. For example, PCA provides explicit information about the principal directions directly, whereas PointHop, like [13], provides the overall principal directions by partitioning the regions and computing the centroids of the regions. However, although LLE explicitly provides the orientation of all neighborhood points, it is difficult to directly extract the overall main orientation, so the property of global shape is missing.

Local Detail: The global shape above provides a simple geometric prior, but the detailed information inside the local region is missing, so we believe that a valid ES should

Table 2. Rich Geometric Information

Method	Relative Coordinates	Normal Vectors	Geodesic Distances
PointNet++	58%	0.26	0.051
PointMetaBase	63%	0.24	0.038
X-3D	**74%**	**0.16**	**0.019**

also have the property of A. PCA uses three eigenvalues and eigenvectors to describe the overall shape, but it lacks information about the internal neighborhood points, so the local detail property is missing.

We believe that LR pays more attention to the local information of each point detail in the local structure, PCA pays more attention to the global information in the local structure, and PH is a better balance between the internal details and the overall shape.

E.2. Reasons

Global Shape provides a good geometric prior for category learning in local regions. This part has been described in detail in the main text.

Local Details provide richer geometric informations. To demonstrate that local details provide richer geometric information for each neighborhood point, we design tasks to predict geodesic distance, normal vector, relative position which is shown in Table 2 , and the performance of these tasks demonstrates the richness of geometric information

We predict those geometric metric of neighborhood points based on the aggregated local structures and neighborhood point features which can be written as follow:

$$Predict = MLP([\hat{\mathbf{f}}_i^{(2)}, \mathbf{f}_{i,j}]) \tag{16}$$

For relative coordinates, direct prediction is difficult, inspired by PointRCNN [37], we adopt a bin-based strategy, that is, the relative coordinates are divided into bins according to the maximum and minimum values, and predict which bin it is in, which turns into a classification task, and we report the classification accuracy of this task.

Then for normal vectors and geodesic distances, we report its MSE loss directly.

Explicit local structure provides strong robustness. In Table 3, we report that X-3D is more robust against simple local transformations (rotation, scale). Inspired by [35], more powerful data augmentation methods such as cutmix can better study the robustness of point cloud models. Therefore, in this part we use a stronger data augmentation technique, through PointCutmix [50], to combine different objects by cutmix opearator to achieve the purpose of distorting and occluding the local structure. In Table 3, we evaluate PointMetaBase and X-3D on ShapeNetPart using PointCutMix and report the mIoU under different cutmix ratioλ. Furthermore, we test two cases. The first one uses objects of the same category for cutmix to simulate the local distortion of objects, and the second one uses objects of different categories for cutmix to simulate the occlusion stacking among different objects. Experimental results demonstrate the robustness of X-3D in the face of various difficult situations in real scenes

Table 3. Robustness

Method	λ=0	λ=0.1	λ=0.2	λ=0.3	λ=0.4
Same Category CutMix					
PointMetaBase	86.7	86.5	84.4	82.7	81.2
X-3D	86.9	86.9	86.7	86.0	85.3
Different Category CutMix					
PointMetaBase	86.7	70.3	51.6	36.6	27.7
X-3D	86.9	80.1	71.8	59.2	46.1

Table 4. We tested the difference between the implicit local structure of the feature Spaces at different layers and the explicit space of the input 3D space on PointMetaBase.

	layer1	layer2	layer3	layer4
GAP	1.36	3.23	5.37	7.02

Explicit structure provides 3D geometric information that is missing from the feature space. Most of the existing works can be classified as implict high-dimensional structure modeling, which leads to more and more serious loss of explicit 3D geometric information with the increasing depth of the model, which is shown in Table 4. Therefore, by explicitly introducing local structure, X-3D supplements the deep feature space with 3D geometric information, which effectively improves the performance