Journal of Imaging Science and Technology® 66(3): 030503-1–030503-13, 2022.

Geometric Encoded Feature Learning for 3D Graph Recognition

Li Han, Pengyan Lan, Xiao-min Wang, Xue Shi, Jin-hai He, and Gen-yu Li
Liaoning Normal University, Huanghe Road 850, Shahekou District, Dalian, Liaoning Province, 116029, China
E-mail: lhan98@lnnu.edu.cn

Abstract. *To meet the requirement of diverse geometric transformation and unstructured graph data for intelligent shape analysis method, this paper proposes a novel 3D graph recognition method based on geometric encoded feature learning, which effectively optimizes the feature extraction process from low-level geometry to high-level semantics, and improves the generalization and robustness of deep learning. Firstly, we adopt GMDS and KNN to build isometric embedding space and extract intrinsic geometric features. Secondly, in combination with the BoF method, the unified geometric encoded feature is generated, which effectively enhances the shape description ability of all kinds of graph data. Finally, an adaptive dynamic graph convolution network is established. Through dynamic spectral graph convolution and weighted feature refining, we implement efficient deep feature extraction and 3D graph recognition. A series of experimental results show that our proposed method achieves better performance in graph recognition and classification task. Moreover, whether the 3D graphs are rigid or non rigid, or incomplete and unconnected graph data, our method is significantly robust and efficient.* © *2022 Society for Imaging Science and Technology.*
[DOI: 10.2352/J.ImagingSci.Technol.2022.66.3.030503]

1. INTRODUCTION

Because traditional 3D shape analysis methods [1–3] rely on the low-level geometric features, it is difficult to get meaningful results when dealing with shapes having complex geometry and topology.

The successful applications of neural networks in image processing, speech recognition and natural language processing have promoted the study of 3D shape analysis based on deep learning model. It is mainly divided into three aspects: handcrafted feature-based method, view-based method and original data-based method [4].

Handcrafted feature-based methods learn high-level features from low-level features. For example, DeepSD [5], SGWC [6], DeepGM [7], etc. However, they are limited in specific topological structure and heavy eigenfunction solving. View-based learning methods convert 3D shape into a series of 2D images, and then extract the view features by deep learning frameworks [8–10], which are efficient since they make full use of the successful CNN learning model in the field of the 2D image. However, view-based learning methods only consider the visual similarity and ignore the intrinsic geometric information between 3D shapes. Recently, the methods based on original data have been widely studied, which can directly learn features from original 3D data. For example, PointNet [11] manipulated points and solve the permutation invariant problem by a multi-layer perceptron (MLP) without considering the local geometry. PointNet++ [12] further applies PointNet recursively on the nested partitions to extract local features and combines learned features from multiple scales. However, PointNet++ still processes each point in the local point set individually and does not extract the relationships. And 3D Shape Net [13], VoxNet [14] and OctNet [15] use CNN to learn features from voxelized 3D shapes.

Although a large number of methods have emerged for 3D data analysis, they are either effective for specific geometry or limited to the global shape analysis while ignoring the local structural similarity.

An increasing number of applications where 3D data are represented in the form of graphs have triggered the research upsurge of graph-based learning system. As graphs can be irregular and may have a variable number of unordered nodes, which has imposed significant challenges on existing machine learning algorithms. Researchers have explored them in two main streams, spectral-based method and spatial-based method. The first is to define and optimize the spectral convolution operator on local graph [16–18] or global graph [18–20]; the other is to gather the node feature from its spatial neighborhood by designing dynamic convolution operator [21, 22], attention weighted networks and neighborhood sampling mechanism so on [23, 24]. However, they still face challenges such as the tedious computation, the generalization of diverse types of graph, the adaptivity to geometric transformation of irregular graph, the performance of large-scale graph learning.

This paper presents a general deep feature learning method for 3D graph data recognition. The two main contributions of this paper are: to propose a general geometric embedding and coding method to effectively measure the intrinsic features of irregular graph data and to present a novel graph convolution network to enhance the adaptability to various geometric transformations.

Firstly, we embed irregular 3D graph data into an isometric space, where the intrinsic geometric features (low-level features) are extracted based on the sparse sampling and generalized multidimensional scaling (GMDS) analysis. Secondly, the bag of feature (BoF) model is used to generate geometric encoded features (middle-level

Received July 16, 2021; accepted for publication Nov. 9, 2021; published online Feb. 4, 2022. Associate Editor: Gabriele Gianini.
1062-3701/2022/66(3)/030503/13/$25.00

features), which are robust to diverse graph structure and geometric transformation. Finally, an adaptive dynamic graph convolution network (ADGCN) is established to effectively learn deep features (high-level features) from geometric encoded features. Experimental results show that the proposed method achieves the effect of state of the art in shape analysis and recognition.

The paper is structured as follows: we present the related works in Section 2. The encoded geometric feature construction is presented in Section 3. Section 4 introduces our adaptive dynamic graph convolution network for 3D data recognition. Experimental results are analyzed and discussed in Section 5. We conclude our work in Section 6.

2. RELATED WORKS

Unlike images, which typically have fixed grid structure, graph data has irregular structure, making it difficult to define convolution and pooling operations. Moreover, graph data can be complex, containing diverse types and numerous properties, and quite often, graphs may have millions of nodes and edges. Great efforts have been made to solve these problems [25, 26].

Bruna et al. [27] first developed a graph convolution based on the spectral graph theory. Spectral graph CNN models have isometry invariance and hence have been applied to non-rigid shape analysis [28]. However, this approach had a number of drawbacks including the computational complexity of Laplacian eigendecomposition, and a lack of spatial localization, as the Laplacian eigenbasis is domain-dependent. Defferrard et al. [17] introduced ChebNet and used a polynomial filter in spectral method, which greatly reduced the time complexity. GraphHeat [29] adopted HKS as convolution kernel to realize the low-pass filter instead of Chebyshev polynomials. DCNN [19] replaced the eigenbasis of the convolution by a diffusion-basis. DGCNN [21] further improved the graph neural network with a dynamic local convolution kernel and spectral clustering. Due to different nearest neighbor graph structures, it is impossible to directly apply the spectral analysis of graph Laplacian. Following this, Yi et al. [30] used SyncSpecCNN to align different graph structures to a standard space in the spectral domain, which has achieved the effect of state of the art. PATCHY-SAN algorithm [20] transformed graph structure into sequence structure, and then directly applied the convolution on the sequence structure by convolution neural network.

Spatial-based Graph convolutional network uses a convolution-like operation to aggregate features of all adjacent nodes for each node, followed by a linear transformation to generate a new feature representation for a given node. Graph attention networks (GATs) [23] employed the attention mechanism to obtain different and trainable weights for adjacent nodes by measuring the correlation between their feature vectors and that of the central node. In neighborhood sampling, GraphSAGE [22] uniformly sampled a fixed number of neighbors for each node during training. FastGCN [31] further adopted a layer-wise sampling in each convolutional layer to reduce variances and lead to better performance. Wang et al. [21] (DGCNN) designed an edge convolution operator to extract feature from a center point and the edge vector from its neighbor to itself. They not only searched neighbors in the input Euclidean space, but also clustered similar features in the feature space. However, DGCNN still has some limitations, such as their neighbors may be too similar to provide valuable edge vectors; or the features are extracted independently without considering the global topology; or there are many trainable parameters when we train the whole network.

In order to better understand and recognize complex and diverse 3D graph data, the shape features should not only be distinctive, but must also be adaptive and stable to various geometric transformations.

Inspired by DGCNN, we provide a general deep learning method for diverse graph data analysis, which not only takes the local geometry into account, but also generates a global encoded feature for any type of 3D graphs. We also design an adaptive dynamic graph convolution network (ADGCN) to improve the stability and robustness of deep feature learning.

Figure 1 illustrates the pipeline of our proposed method. Firstly, we build an isometric embedding space for graph data where the embedded geometric features of HKS and distance are extracted. Secondly, we generate a compact and uniform representation by encoding the geometric features into global statistic feature, which has stronger discrimination and stability. Finally, an adaptive dynamic graph convolution network (ADGCN) is established to implement efficient deep feature learning and graph recognition via dynamic spectral filtering and weighted feature refining. Experiments demonstrate that our learning method achieves better results than the state-of-art on classification tasks.

3. GEOMETRIC FEATURE EMBEDDING AND ENCODING

3D data represented by irregular graph includes connected graph and disconnected graph. Usually the spectral convolution is adopted for the connected graph learning, such as: GCN [17], DGCNN [21]. However, due to the large scale of nodes; it involves in the tedious Laplacian eigendecomposition and the huge storage problem. For the disconnected graph data, it is often considered as point clouds and extracts geometric features through local point set and multi-layer perception (PointNet, PointNet++). But it neglects either the local geometric connectivity or the global topology.

In order to effectively obtain the discriminative features for general 3D graph data, we adopt GMDS [32] to convert irregular graph data into an isometric embedding space and construct geometric encoded feature with the help of bag of features, which can effectively improve the generalization ability and robustness of shape descriptor.

3.1 *Isometric Embedding Based on GMDS*

The core idea of GMDS [32] is to implement the isometric shape embedding and matching by calculating the minimum

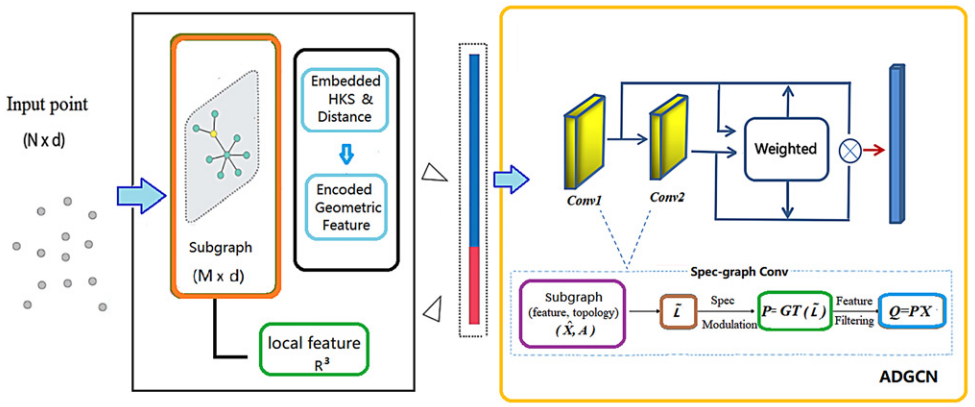


Figure 1. The framework of our method (taking 3D coordinates and their local neighborhood as input, we map them into an isometric space to generate embedded features based on GMDS. Then combining with the embedded HKS and the embedded distance, the BoF model is adopted to construct geometric encoded features, which can better reveal global structural properties of diverse graph data, the local topology and the geometric encoded features are input into an ADGCN to learn the discriminative deep features for classification task).

distortion mapping on the sampled surface in the finite metric space. This mapping function transforms the non rigid matching problem into the rigid matching problem effectively by embedding the geometric structure of the surface into the low dimensional Euclidean space. We make full use of the advantages of GMDS, and propose an optimized embedding method, which can effectively embed complex, large-scale transformation graph data into a unified metric space.

3.1.1 *Sampled Surface*

If a surface S is covered by a set of surfaces $S^r \subset S$, then the set $S = \cup_{s \in S^r} B_S(s, r)$ is called "r-covering" of S, where $B_s(s_0, r) = \{s \in S : d_S(s, s_0) < r\}$ is the sphere with radius r around the point S_0 in S. A finite r-covering of S consisting of N points is denoted by S_N^r. The metric on S_N^r is assumed to be the restricted metric $d_S \mid S_N^r(s, s') = d_S(s, s')$; for all s, s' in S_N^r. An arbitrary finite sampling of S consisting of N points is denoted by S_N, where S is called "continuous" surface and S_N a "discrete" one.

Given two surfaces S and Q, a transformation $\psi : Q \to S$ is said to have "distortion" ε, that is dis $\psi \equiv \sup |d_Q(q, q') - d_S(\psi(q), \psi(q'))| = \varepsilon$. Then the transformation $\psi : Q \to S$ with dis $\psi = 0$ is called an "isometry".

Assume two sampled surfaces S_N^r and $Q_{N'}^r$ (representing the continuous surfaces S and Q) and the geodesic distances between the samples represented by an $N \times N$ matrix $D_{S_N} = (d_S(s_i, s_j))$ and an $N' \times N'$ matrix D_{Q_N}, respectively. Each sampled surfaces is mapped into an m-dimensional Euclidean space by a near-isometric embedding, obtained by minimization of the stress function:

$$\arg\min \sigma(X; D_{S_N}) = \frac{1}{N} \sum_{i>j} (d_R m(x_i, x_j) - d_S(s_i, s_j))^2. \tag{1}$$

Here X denotes a $N \times m$ matrix of coordinates in R^m, and d_{R^m} denotes the Euclidean metric. The stress can be thought of as a L_2 measure of the distance distortion caused by such an embedding.

3.1.2 *GMDS*

Similar to the Euclidean case, the "generalized stress" (GMDS) is defined as

$$\arg\min(U; D_{Q_N}, d_S, W) = \frac{1}{\sum_{j>i} w_{ij}} \sum_{j>i} (w_{ij}(d_S(u_i, u_j) - d_Q(q_i, q_j))^p)^{\frac{1}{p}}, \tag{2}$$

where the matrix $\boldsymbol{U}$ represents the positions of N points on S in some local or global parametric coordinates u_i, and $W(w_{ij})$ is a symmetric matrix of nonnegative weights.

The GMDS method adopted "canonical forms" (CFs) based on Euclidean embeddings and can be performed in a computationally efficient manner. The distance measure is effective for simple surfaces, but it is not enough for complex and disconnected 3D graph data.

In this paper, we introduce an isometric embedding metric based on local heat kernel signature (HKS) and the measurement of diffusion distance. Different from GMDS, it does not rely on the graph connectivity and avoids the tedious eigendecomposition, instead, it finds a good way to transform local features to global ones. The extracted shape features not only reveal the consistency of the intrinsic structure of the non-rigid transformation, but also have good generality and flexibility in the connected graphs and disconnected graphs.

First, we calculate the discrete HKS feature $u(s_i)$ for each node by applying discrete Laplacian operator in its K-nearest neighborhood (KNN) [33]:

$$L = A^{-1} W_h; \quad W_h = \frac{1}{4\pi^2} e^{\frac{d(v_i, v_j)}{4t}}, \tag{3}$$

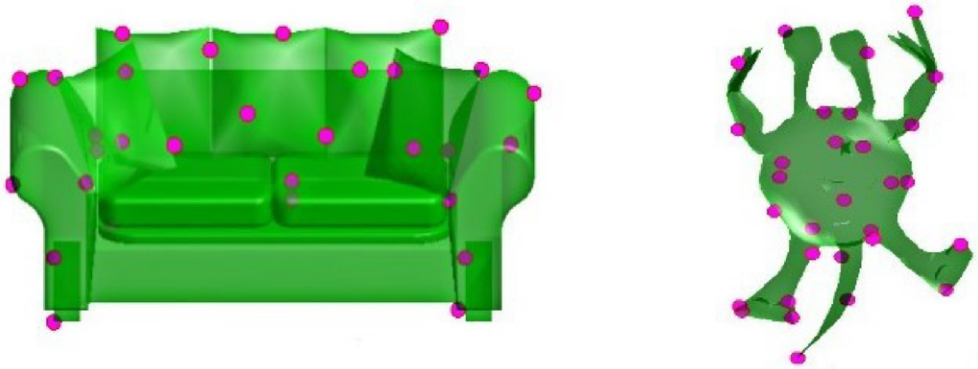


Figure 2. Sampling points based on farthest point sampling (FPS) method on different graphs.

where, W_h is the similarity matrix in the local *KNN* neighborhood, $d(v_i, v_j)$ denotes the distance between any pair of mesh vertices x_i and x_j, and t is a positive, that determines the width of the kernel. $\boldsymbol{A}$ is a positive diagonal matrix, corresponding to the triangle area of the shared node in the graph. The local heat kernel features can be obtained by solving the Laplacian matrix $\boldsymbol{L}$ [34].

Second, we select sampling points $(s_1, s_2, \ldots, s_j)$ based on FPS (see Figure 2), and compute the diffusion distance between the sampling points as parameterized embedded coordinates $d_u(\cdot)$:

$$\arg\min \sigma(X, D_{uN}) = \sum_{i>j} \|d_{R^l}(x_i, x_j) - d_u(u(s_i), u(s_j))\| \in R^{n\times l}\, x_i \in S, s_j \in S_N^l, \tag{4}$$

where an $l \times l$ matrix $D_{uN} = d_u(u(s_i), u(s_j))$ represents the embedding space with diffusion distance between the ith sampling point and the jth sampling point. Then each pair of nodes (x_i, x_j) in Euclidean space are transformed into l-dimension embedding space. In this way, an l-dimensional embedded HKS for each node is constructed.

Our method implements the multidimensional scaling by using discrete embedded HKS features and Euclidean distance measurement instead of using geodesic distance in traditional GMDS; it not only effectively avoids the dependence on topological connectivity, but also better reveals the intrinsic structure of graph data (Figure 3a).

When the parametric coordinates $u(s_i)$ is defined by local distance, we can extend our method to compute embedded distance feature. We, first calculate the average distance $\tilde{D}_i$ for each node s_i with its K-nearest neighborhood $Neb(s_i)$, and then construct l-dimensional distance feature d_g with Euclidean measurement by mapping local average distance to global topology (Fig. 3b):

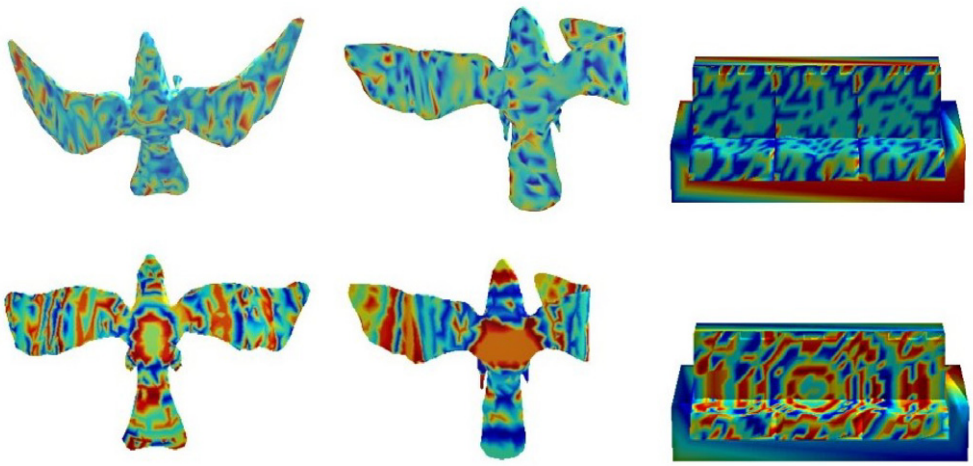
$$\tilde{D}_i = \frac{1}{K} \sum_{s_j \in Neb(s_i)}^{K} d(s_i, s_j); \qquad d_g(s_i) = \|\tilde{D}_{i1}, \tilde{D}_{i2}, \ldots, \tilde{D}_{il}\|. \tag{5}$$

In our experiment, we set the K-nearest neighborhood $K = 8$, the number of sampling point $l = 150$. We use a pigeon (connected graph) in SHREC2011 and a sofa (disconnected graph) in ModelNet10 as examples. The comparison between local HKS (top) and GMDS embedded HKS (bottom) of two pigeons is shown in Fig. 3(a). It can be seen that GMDS embedded HKS features have stronger structure perception ability. Fig. 3(b) illustrates the stability of our embedded distance features between two different types of graph data. It can be seen that the embedded l-dimensional distance feature effectively reveals global topology.

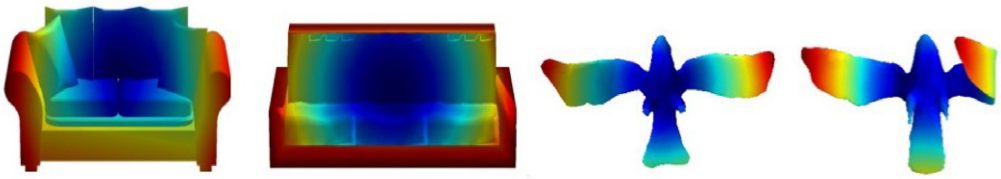
Our embedded features are based on multi-scale geometric measurement in finite sampling space. In order to verify the difference between our embedding space and continuous space, we further compare our embedded features of HKS and distance with the ground truth features. As shown in Fig. 3(c), that two feature curves of pigeon models are extremely similar to the ground-truth feature curves. It confirms that our isometric mapping function effectively reveals the intrinsic geometric features with minimum geodesic error. Meanwhile, our isometric embedding method effectively avoids the time complexity of global affinity matrix construction and large eigendecomposition in massive point cloud data, and thus greatly improves the operation efficiency.

Although our embedding function generates isometric feature for 3D graph data with arbitrary structure (connected or disconnected), it is still low-level geometric feature and unstable to complex topology and noisy data. Therefore, it is very important to reveal intrinsic shape features of large-scale complex and incomplete 3D shapes.

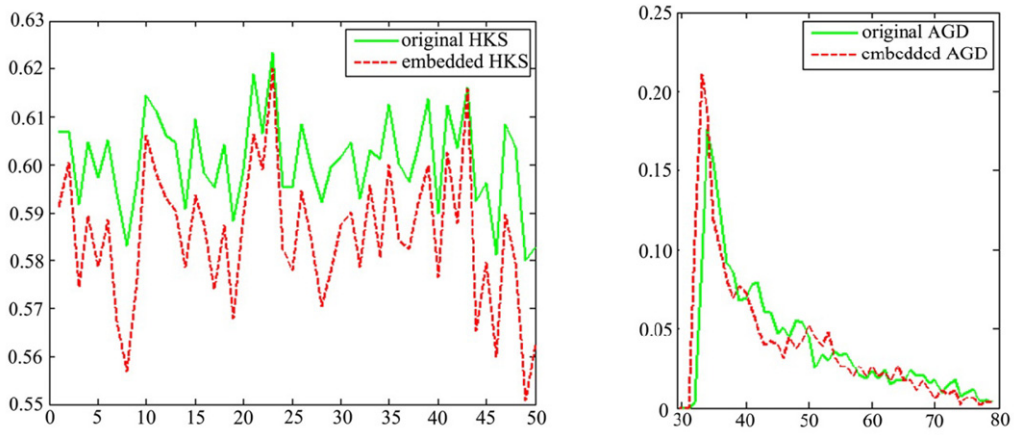
BoF model proposes a good way to create visual vocabulary space, which effectively maps multiple geometric features to the frequency of visual words in order to represent the global distribution of features. This method not only effectively improves the discrimination of shape features, but also greatly improves the robustness of isometric and non-isometric transformation [35–38]. Combining with BoF



(a) GMDS embedded HKS (bottom) represents better geometric details than Local HKS (top)



(b) embedded distance features reveal the consistence of topological structure of non-rigid transformed shapes



(c) original HKS feature and embedded HKS feature (left) of pigeon; original geodesic distance and embedded distance (right) of pigeon (the x-axis represents the sampling points and the y-axis represents the scalar value of the feature)

Figure 3. Embedded features of different models.

model we further introduce a geometric feature encoding method.

3.2 *Geometric Feature Encoding*

Taking the GMDS embedded features as input $f_{in}(S(s_g, s_h))$, where S_g is the embedded distance, and S_h is the embedded HKS, we convert each descriptor $f_{in}(S_i)$ into a vocabulary space, where a codebook is constructed by quantizing it into a k number of codewords. These codewords are usually defined as the centers $V = (v_1, v_2, \ldots, v_k)$, which are generated by an unsupervised K-means algorithm. Each descriptor is then mapped to a codeword in the codebook via the $k \times n$ cluster soft-assignment matrix $\boldsymbol{C} = \{c_{r1}, c_{r1}, \ldots, c_{rn}\}$: The frequency of each word is counted as the output feature. Finally, combining with the local coordinates of each point, we can obtain the geometric

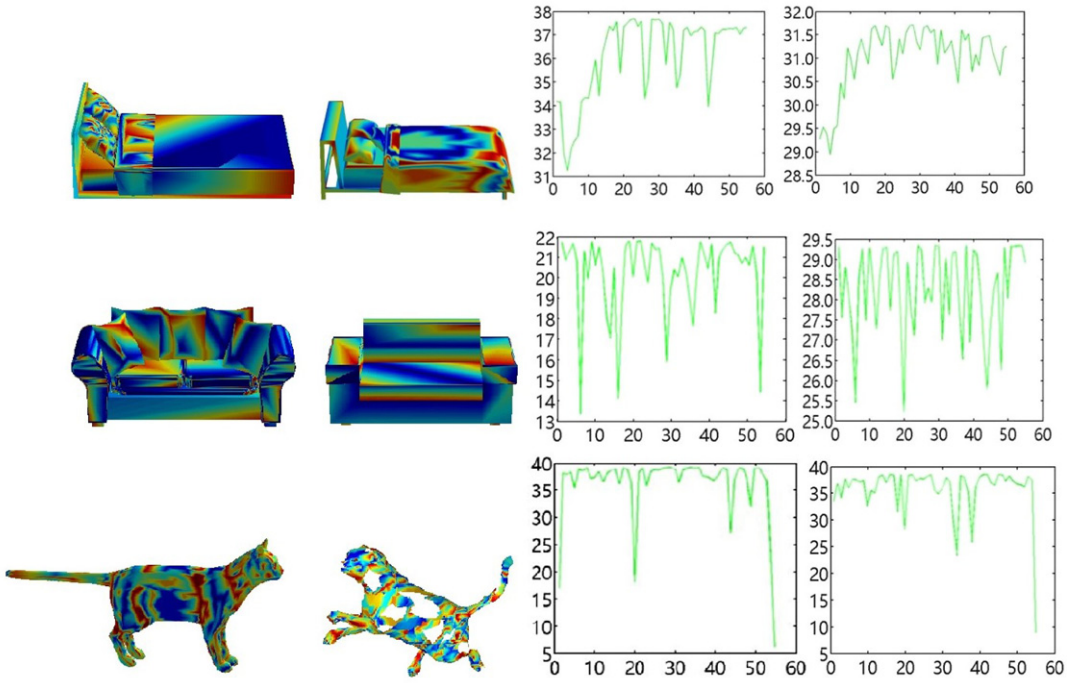


Figure 4. Geometric encoded features of different models (the x-axis represents the number of codewords $k = 55$; the y-axis represents the frequency of each codeword, the color from blue to red denotes the feature value from low to high).

encoded features of the model $f_{\text{geo}}(S) \in R^k$.

$$C_{ri} = \frac{\exp(\alpha \|s_i - v_t\|_2^2)}{\sum_{t=1}^{k} \exp(-\alpha \|s_i - v_t\|_2^2)}; \quad f_{\text{geo}}(S) = \sum_{i=1}^{N} c_{ri}, \tag{6}$$

where $\mathbf{C}$ is denoted as encoded geometric descriptor. $\|\cdot\|_2^2$ is the L_2 -norm, and α is a smoothing parameter that controls the softness of the assignment.

Figure 4 shows the encoded geometric features of different shapes, the curve represents the frequency of each word. In our experiments, we choose the clustering number $k = 55$, it can be seen that the geometric embedded features are discriminative and stable. Especially for the incomplete model, it can effectively reveal the structural consistency.

4. ADAPTIVE DYNAMIC GRAPH CONVOLUTIONAL NETWORK FOR GRAPH RECOGNITION

The encoded geometric feature is structure-aware and has significant representation ability in isometric transformation shapes and incomplete shapes. However, the 3D graph data is complex and diverse; it is still an important issue to explore consistent feature representation for massive, large-scale and non isometric 3D shapes.

Based on the geometric encoded features and the DGCNN framework [21], we propose an adaptive dynamic graph convolution network (ADGCN) for irregular 3D graph data analysis, which effectively improves the efficiency, robustness and versatility of 3D graph recognition (as shown in Figure 5).

The traditional GCN method usually uses average weighted aggregation of adjacent nodes for feature extraction. Due to the different number of adjacent nodes in 3D graph data, the different size of reception field and different weighted filters are needed. DGCNN improves it with an EdgeConv, which captures local geometric structure by constructing a dynamic local neighborhood graph and applying convolution-like operations on the edges connecting neighboring pairs of points.

Inspired by DGCNN, we propose an adaptive dynamic graph convolution network for graph data learning. It takes the geometric encoded features and the local topology as input, applies spectral convolution operator to local graph instead of "EdgeConv", which fully considers different local geometry and can better reveal the local intrinsic properties. The K-nearest local graph is dynamically updated layer by layer in the feature space, which improves the adaptability of the learning process. Meanwhile, a weighting module is added to fuse the output feature of each layer to generate informative deep feature, as it effectively enhances the discrimination of deep feature.

Our learning framework consists of two parts. First, we generate the geometric encoded features for each node based on its local subgraph (k-nearest neighborhood) as mentioned in Section 3:

$$f_{\text{geo}} = g(X_s, A_s, \hat{k}), \tag{7}$$

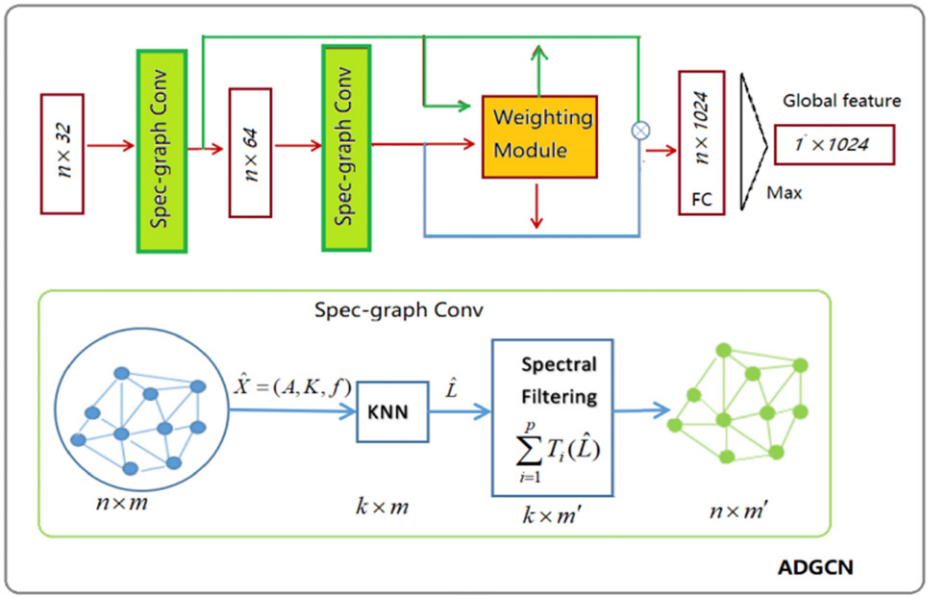


Figure 5. Adaptive dynamic graph convolution network (ADGCN) (ADGCN applies spectral convolution operator on local graphs, where the *KNN*-local graph is dynamically updated in feature space layer by layer. We first take the *KNN*-local graph and the geometric encoded feature (A, K, f) as input, and adopt Chebyshev polynomials as convolution operator to extract local features, the output features of each layer are weighted and fused to generate the final deep features for classification task).

where $g(\cdot)$ is the encoded feature generator, $\mathbf{A}_s$ is the affinity matrix of subgraph, $X_s \in \Re^3$is the sth input node, taking $\hat{k}$ sampling points as input, the geometric encoded feature f_{geo} is extracted by generator $g(\cdot)$.

Second, taking the geometric encoded feature f_{geo} and the local topology as input, the spec-graph convolution operator updates the feature of each node through adjacent nodes and feature filtering:

$$\hat{X}_{l+1} = \sigma(\mu(\cdot)\hat{X}_l W_l), \tag{8}$$

where $\hat{X}_l$ and $\hat{X}_{l+1}$ represent the input and the output of features, W_l is the trainable weight matrix and $\sigma(\cdot)$ is the ReLU function. We use Chebyshev polynomials to simplify the spectral convolution kernel $\mu(\cdot)$:

$$\mu(\hat{A}, K) = \sum_{i=1}^{p} T_i(\hat{L}), \tag{9}$$

where $\hat{L} = 2L/\lambda_{\max} - I$, L is the Laplace-Beltrami operator of subgraph (X, A, K).

Our ADGCN is built with two convolutional layers, two pooling layers and one full connection layer (FC) (see Fig. 5). There are 32 convolution kernels in the first layer and 64 convolution kernels in the second layer. Each layer uses the ReLU activation function and the max-pooling operation. We take the local topology $\mathbf{A}_s$ and geometric encoded feature f_{geo} as input in the first layer, and apply spectral convolution to extract deep feature from its neighborhood; furthermore, we update the local graph by recomputing the K-nearest neighbors in the feature space, and then input the renewed local graph and features into second layer. The output features of each layer are weighted and concatenated. Finally, the deep features are fed into a Softmax classifier for recognition task. We take cross-entropy loss function to optimize the learning process.

The deep learning algorithm is summarized in Algorithm 1.

5. EXPERIMENTS

We conducted extensive experiments to validate the effectiveness of our method on public SHREC and ModelNet datasets. Among them, 60% of the models are randomly selected as training sets, 20% models as test sets, and 20% models as verification sets.

- SHREC is a standard3D Shape Retrieval Contest (SHREC) datasets which includes SHREC-2010, SHREC-2011, SHREC-2015 and SHREC-2016. SHREC10 contains 200 non-rigid models with different postures from 10 classes. SHREC-2011 consists of 600 mesh models from 30 categories. SHREC-2015 is a dataset of 1200 watertight mesh models with 50 classes. SHREC-2016 contains 400 incomplete models with cuts and holes.
- ModelNet includes two datasets which respectively contain CAD models of 10 and 40 categories. ModelNet10 consists of 4899 object instances. ModelNet40 consists of 12,311 object instances.

Algorithm1. Geometric encoded feature learning for graph data analysis

Input: Training Dataset $M = \{M_1,...,M_m\}$ 3D graphs

$\boldsymbol{B}$ (b =1,2,…., c) classes and unknown test model $\hat{y}$.

Output: 1024-dimensional deep feature and the predicted class label $P(c \mid \hat{y})$ for unknown test model.

Step1.Geometric encoded feature Generator

1.1 For $i = 1$ to $\hat{n}$ do // $\hat{n}$ is the number of training data, and n is the number of nodes of each graph //

1.2 Compute l sampling points based on FPS

1.3 Compute GMDS embedded geometric features (HKS, distance) $f_i(S(s_g^i, s_h^i))$ for each graph M_i ;

1.4 Apply k-means algorithm and find the $k \times n$ middle-level feature matrices C_A^i based on BoF model ,

1.5 Generate k-dimensional geometric encoded feature $f_{geo}(S_i)$

1.6 End for

Step2. Construct adaptive dynamic graph convolution network for 3D graphs learning

while ($i \le T$) do

2.1 Input training data into ADGCN

2.2 Feature learning: $F_i = \sigma(\mu(\cdot), g(\cdot), W)$ where $\mu(\cdot)$ is the Chebyshev polynomial convolution kernel on local graph, $g(\cdot)$ is geometric encoded feature, $\boldsymbol{W}$ is learned weight matrix, F_i is the output deep feature of model M_i

2.3 Compute the accuracy of input Model M_i in Softmax classifier: $\hat{p}(F_i)$ and $Acc(i)$

2.4 Cross entropy loss optimization: i++;

Endwhile

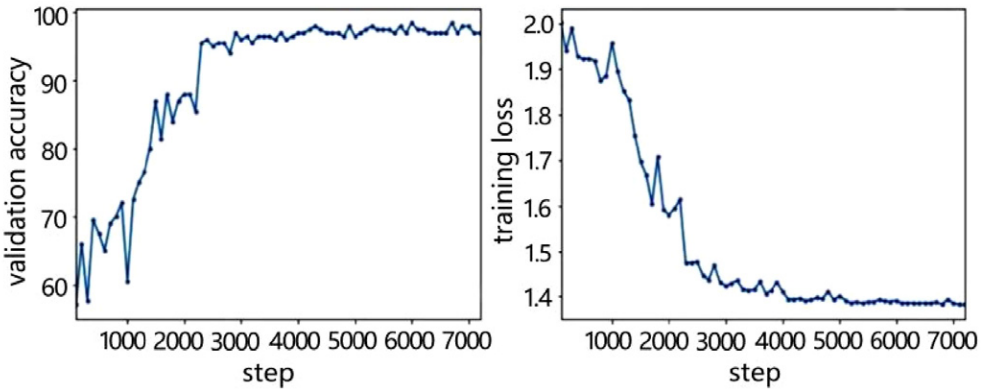


Figure 6. Training accuracy and loss curves on ModelNet40.

In our experiment, we set the K-nearest neighborhood to 8, the l sampling point is 150, an initial learning rate is 0.001 and the learning attenuation rate is 0.95, the momentum is 0.9, and the polynomial coefficient is 10. Batch size is 16. All layers are implemented with batch normalization.

Figure 6 shows the training process of our model in ModelNet40. It can be seen that when the number of iterations reaches 2500, the accuracy and loss curve tends to be stable. Therefore, in our experiments, we set the number of iterations to 3000 to obtain the optimal training results.

In this section, we first discuss the time complexity of our method, and then evaluate the performance under different sampling points. Furthermore, we verify the robustness and stability of our geometric encoded feature and deep feature based on ADGCN learning model. Finally, we comprehensively analyze the advantages and limitations of our method against other state-of-the-art methods.

5.1 *Time Complexity of Our Method*

Our work mainly includes two modules. One is geometric feature extraction module and the other is deep feature learning module. In geometric feature extraction module, we first calculate the discrete HKS of each node according to its *k-nearest neighborhood*. Secondly, l sampling points are chosen by using FPS, and l-dimensional embedded HKS is generated through diffusion distance measurement. The time complexity is $O(\ln K^2)$, where n is the number of nodes, K is the average number of local neighborhood of

Table I. Time complexity and accuracy analysis of different methods on ModelNet40.

Method	Infer/ms	Acc (%)
PointNet [11]	25.3	89.2
PointNet++ [12]	163.2	90.7
DPAM [40]	36.6	91.4
DGCNN [21]	27.2	92.9
Ours	24.8	92.7

Table II. The classification performance (average accuracy %) by using different features on SHREC-2015, SHREC-2016, ModelNet10.

Dictionary	SHREC 2010	SHREC 2011	SHREC 2015	Mean in (SHREC)	ModelNet10
Embedded HKS	86.34	85.80	82.40	84.85	90.53
Geometric Encoded Feature	95.41	97.23	94.24	95.63	92.36
Deep Feature based on ADGCN	98.19	98.72	97.94	98.28	97.14

each node, and l is the sampling points over the shape. Because our method computes the discrete HKS based on local neighborhood, it effectively avoids the tedious computation caused by eigendecomposition on the whole graph, Furthermore, we adopt BoF model to transform the low-level shape descriptor into a vocabulary space, and construct geometric encoded features via k cluster and soft-assignment matrix. It costs $O(kn^2)$, k is the number of clustering center, which is set to 55 in our experiment. Therefore, the overall time complexity of geometric feature extraction is $O(\ln K^2) + O(kn^2)$, since K, l, k are much less than n, it thus costs $O(n^2)$. In deep learning module, we take the geometric encoded feature and local topology as inputs and learn the deep features by a two-layer weighted spectral convolution network, where the *KNN* local graph is recomputed based on the features of each spectral convolution layer. We weight the feature of each layer and concatenate them to generate a 1024 dimensional feature as output. Table I shows that our learning model achieves the better trade-off between the computational complexity and the resulting classification accuracy.

5.2 *Number of Sampling Points*

First, we evaluate the influence of sampling points on geometric feature encoding and shape classification; we sample 30, 50, 80, 100, 120 and 150 anchor points, respectively. As shown in Figure 7, with the increase of the number of sampling points, the accuracy is improving. Taking the ModelNet10 as an example (Fig. 7a), when the number of sampling point is 30, the experimental accuracy reaches 89.85%, and when 150 sampling points are taken, the accuracy reaches 97.14%, an improvement of 7.29%. The experimental results on different datasets have shown the consistence in Fig. 7(b). In order to balance accuracy and efficiency, we use 150 sampling points to extract embedded shape features in the following experiments.

5.3 *Robustness of Geometric Encoded Feature*

Second, we test our geometric encoded feature on diverse graph data. As shown in Figure 8 whether they are complex disconnected graphs, or non-rigid deformed graphs and incomplete graphs, the geometric encoded features provide stable and discriminative representation, revealing both the intrinsic structural consistence between the same category and the significant discrimination between the different kinds of graphs. Specifically, it is robust to the incomplete graph data.

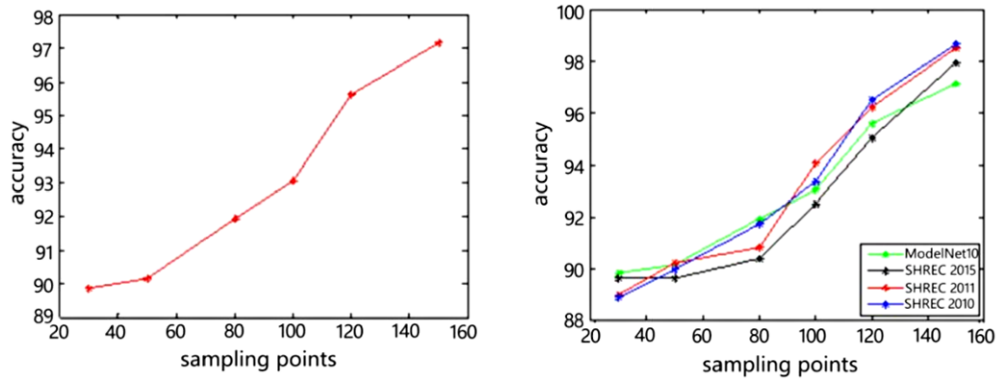
5.4 *Performance of Deep Feature*

Our method builds a bridge from low-level geometric features to high-level semantic features for general graph data, which constructs encoded feature based on local geometry embedding, and then extracts deep features based on our ADGCN learning model. In order to study the performance of our method, we compare the embedded feature, geometric encoded feature and deep feature in Table II.

We feed different features into a standard Softmax classfier, and evaluate the classification performance based on standard 3D shapes benchmarks. We can see that the classification result of geometric encoded features is 1.83% higher than embedded features on Modelnet10 dataset, the average improvement is 10.78% on SHREC dataset. While the deep feature based on our ADGCN is better than that of encoded feature, and the average improvement rate is 2.65% and 4.78%. The classification results of each category with these three features on different datasets are shown in Figure 9.

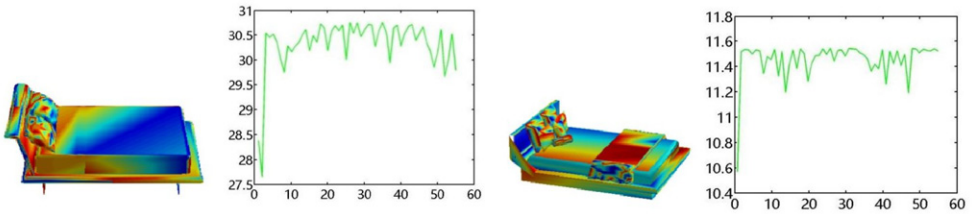
We further compare our proposed method with several baseline methods on three benchmarks in Table III.

First, we compare our approach against the classic handcraft feature-based deep learning methods, such as GA-BoF [37], SGWC [6], DeepGM [7] and DeepShape [36]. GA-BoF only used scale invariant heat kernel and average geodesic distance to generate middle-level descriptor to classify 3D shapes, so it has the lowest performance. SGWC used spectral graph wavelet signatures to construct middle-level features and implement the classification task by multiclass SVM. While DeepGM and DeepShape adopted geodesic moments and multiscale HKS as input, and then learned deep features based on auto-encoder and deep belief networks (DBN). We can see that the classification accuracy of our deep feature is 2.53% higher than that of DeepGM, and 3% than that of SGWC and DeepShape in SHREC datasets. Unlike DeepGM, DeepShape and SGWC, which rely on geometric connectivity and global similarity matrix, our method creates geometric encoded features by local embedding and visual vocabulary mapping, which has stronger generalization and discrimination ability in variety of graph data.

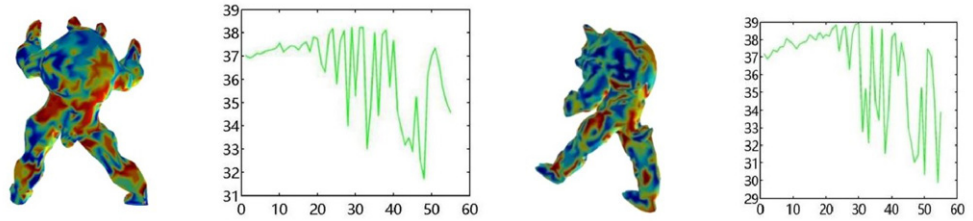


(a) experimental results based on ModelNet10 (b) experimental results on different datasets

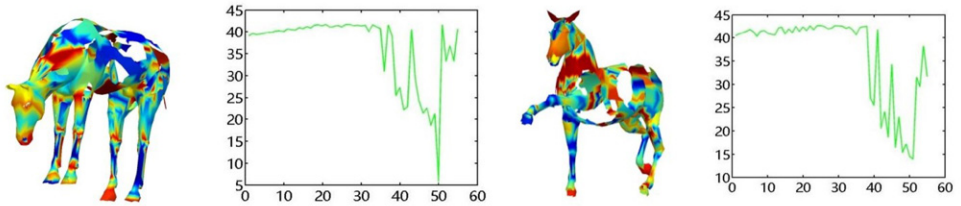
Figure 7. Recognition accuracy based on different number of sampling points.



(a) geometric encoded features of disconnected graphs in ModelNet10



(b) geometric encoded feature of non rigid deformed graph data in SHREC2015



(c) geometric encoded features of incomplete graph data in SHREC2016

Figure 8. Geometric encoded features of different graph data (The *x*-axis represents the number of cluster center (codewords); the *y*-axis represents the frequency of each codeword, the color from blue to red denotes the feature value from low to high).

Second, we made a contrastive analysis between our method and state-of-art deep learning methods, such as LP-3DCNN [39], DPAM [40], PointwiseNet [41], SK-Net [42] and DGCNN [21]. LP-3DCNN uses ReLPV module to extract feature graph from each 3D local neighborhood of input graph. Then, after passing through the activation function, these feature maps at different frequency points are combined linearly. DPAM inserts three point modules into the framework, and combines multi-level features to improve the representation ability. DGCNN presented an EdgeConv, which extracts edge features by establishing KNN local graph, and provides a dynamic convolution mechanism to effectively improve the representation ability of deep features. PointwiseNet aggregates the low-level geometric descriptors of each point with some visual words to indirectly express its high-level semantic information. SK-Net proposed an end-to-end network framework to jointly optimize the inference of spatial key point with the learning of feature

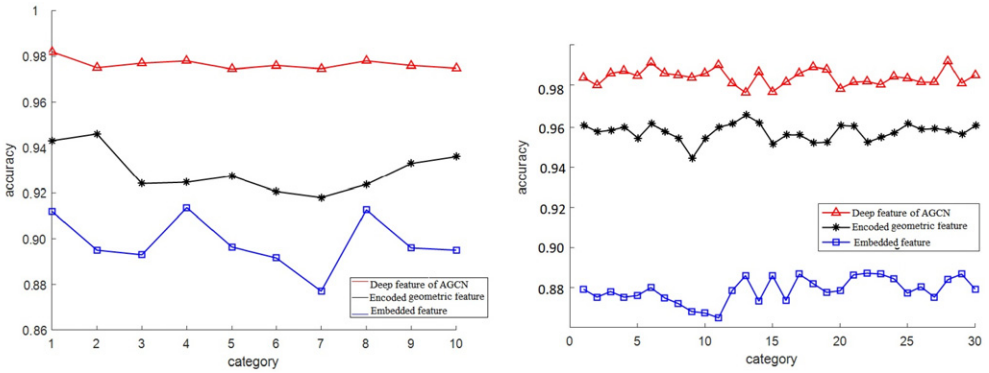


Figure 9. Classification results of each category on ModelNet10 (left) and SHREC2011 (right).

Table III. Comparison of different methods on three datasets (Average accuracy %).

Method	SHREC 2010	SHREC 2011	SHREC 2015	Mean in (SHREC)	ModelNet10
GA-BoF [37]	86.02	93.20	72.93	84.05	—
DeepShape [36]	**95.5**	**96.53**	**92.87**	**94.97**	—
SGWC-BoF [6]	**95.66**	**97.66**	**92.54**	**95.29**	—
DeepGM [7]	**96.33**	**97.89**	**93.03**	**95.75**	—
DGCNN [21]					**94.0**
LP-3DCNN [39]	—	—	—		**94.3**
DPAM [40]	—	—	—		**95.1**
PointwiseNet [41]	—	—	—		**95.5**
SK-Net [42]	—	—	—		**96.2**
Ours	**98.19**	**98.72**	**97.94**	**98.28**	**97.14**

representation of a point cloud for a specific point cloud task. The key the network is to generate Skeypoints and then extract local structure and normalized spatial pattern based on key points.

As we can see that our proposed method effectively optimize the process of geometric feature coding to deep feature learning, and the average classification accuracy on ModelNet datasets is 97.14%, which is 0.94% higher than SK-Net and 1.64% higher than PointwiseNet. Our deep feature enhances the discrimination and robustness to diverse graph data and achieves better classification accuracy, which is about 3% than LP-3DCNN and DPAM. In particular, it provides a general deep learning platform for both CAD data and non-rigid transformed data and incomplete data.

We test the precision-recall curves of our proposed method on SHREC 2011 and ModelNet10 (see Figure 10). Compared to the typical handcrafted feature based methods, GA-BoF has the lowest precision, since it only used the geometric features. SGWC and DeepGM achieve comparable results which learn the high-level feature from multiscale wave kernel signature and geodesic moments. Whether it is handcrafted-based feature extraction method or deep learning method, our method has achieved better retrieval performance.

6. CONCLUSION

This paper presents a generalized deep learning method for irregular 3D graph, which provides a work-flow from local feature embedding to multi-feature encoding and deep feature learning. The extensive experimental results have shown its efficiency and effectiveness for various 3D data. The other valuable part of our method is to improve the generalization and discrimination via an adaptive dynamic graph convolutional network, which achieves significantly better performance than state-of-the-art methods.

However, with rapid increase of large-scale graph data, there will be huge time complexity in the process of local feature embedding and geometric feature encoding, it will be an inevitable trend to explore more efficient graph-based deep learning model for large-scale graph data. Recently, many significant works have been proposed around reducing model size, enhancing the anti-inference ability of training process, improving augmented and unsupervised learning model so on [43–45].

In the future work, we will further extend our method to optimize our learning model, improve the learning process, automatically extract structural features for semantic segmentation and shape understanding.

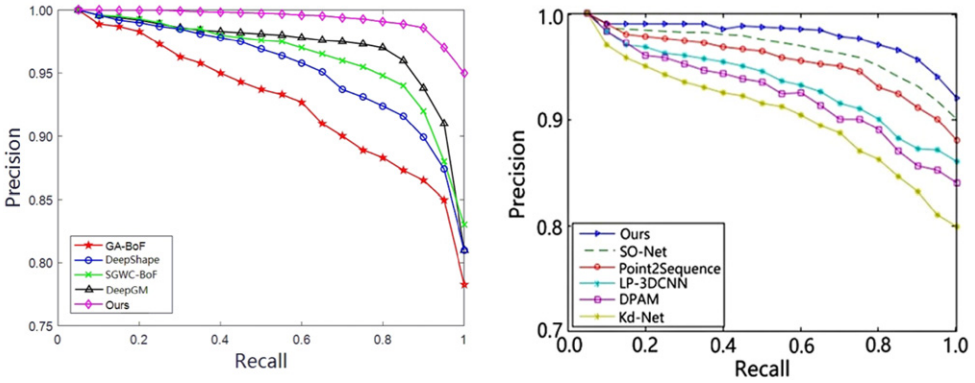


Figure 10. Comparison of PR curves of different methods on SHREC2011 and ModelNet10.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their helpful comments. The research presented in this paper is supported by a grant from NSFC (61702246), a grant from research project of Liaoning province (2020JH4/10100045, LJ2020015) and a fund of Dalian Science and Technology (2019J12GX038).

REFERENCES

[1] R. M. Rustamov, "Laplace-Beltrami eigenfunctions for deformation invariant shape representation," *Proc. SGP* (Eurographics Association, Aire-la-Ville, 2007), pp. 225–233.

[2] M. Ovsjanikov, A. Bronstein, M. Bronstein, and L. J. Guibas, "Shape Google: A computer vision approach to isometry invariant shape retrieval," *Proc. IEEE 12th Int'l. Conf. Comput Vis Workshops* (IEEE, Piscataway, NJ, 2009), pp. 320–327.

[3] C. Li and A. B. Hamza, "A multiresolution descriptor for deformable 3D shape retrieval," User Model. User-Adapt. Interact. **29**, 513–524 (2013).

[4] W. Mohamed and A. B. Hamza, "Deformable 3D shape retrieval using a spectral geometric descriptor," Appl. Intell. **45**, 2213–2229 (2016).

[5] Y. Fang, J. Xie, G. Dai, M. Wang, Z. Fan, T. Xu, and E. Wang, "3D deep shape descriptor," *Proc. 28th IEEE Conf. on CVPR* (IEEE, Piscataway, NJ, 2015), pp. 2319–2328.

[6] M. Masoumi, C. Li, and A. B. Hamza, "A spectral graph wavelet approach for nonrigid 3D shape retrieval," Pattern Recognit. Lett. **83**, 339–348 (2016).

[7] L. Luciano and A. B. Hamza, "Deep learning with geodesic moments for 3D shape classification," Pattern Recognit. Lett. **105**, 182–190 (2017).

[8] B. G. Shi, S. Bai, Z. C. B. Zhou, and X. Bai, "DeepPano: deep panoramic representation for 3D shape recognition," IEEE Signal Process. Lett. **22**, 2339–2343 (2015).

[9] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," *Proc. IEEE Int'l. Conf. on Computer Vision (ICCV)* (IEEE, Piscataway, NJ, 2015), pp. 945–995.

[10] H. Guo, J. Wang, Y. Gao, J. Li, and H. Lu, "Multi-view 3D object retrieval with deep embedding network," IEEE Trans. Image Process. **25**, 5526–5537 (2016).

[11] C. R. Qi, H. Su, K. C. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2017), pp. 77–85.

[12] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: deep hierarchical feature learning on point sets in a metric space," Adv. Neural Inf. Process. Syst. **30**, 5099–5108 (2017).

[13] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, Z. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," *Proc. CVPR* (2015), pp. 1912–1920.

[14] D. Maturana and S. Scherer, "VoxNet: a 3D convolutional neural network for real-time object recognition," *Proc. IEEE/RSJ Int'l. Conf. on Intelligent Robots and Systems (IROS)* (IEEE, Piscataway, NJ, 2015), pp. 922–928.

[15] G. Riegler, A. O. Ulusoy, and A. Geiger, "OctNet: learning deep 3D representations at high resolutions," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2017), pp. 6620–6629.

[16] T. N. Kipf and M. Welling, *Semi-Supervised Classification with Graph Convolutional Networks* (Cornell, New York, USA, 2016).

[17] D. Michael, B. Xavier, and V. Pierre, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in Neural Information Processing Systems* (Curran Associates Inc., Red Hook, NY, USA, 2016), pp. 3844–3852.

[18] A. Micheli, "Neural network for graphs: A contextual constructive approach," IEEE Trans. Neural Netw. **20**, 498–511 (2009).

[19] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," *Proc. NIPS* (Curran Associates Inc., Red Hook, NY, USA, 2016), pp. 1993–2001.

[20] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," *Proc. ICML* (PMLR, New York, USA, 2016), pp. 2014–2023.

[21] 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** (2018).

[22] W. L. Hamlton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Proc. NIPS* (Curran Associates Inc., Red Hook, NY, USA, 2017), Vol. 30, pp. 1025–1035.

[23] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *Proc. 7th Int'l. Conf. on Learning Representations* (Cornell, NY, USA, 2018), p. 10903.

[24] J. B. Lee, R. A. Rossi, S. Kim, N. K. Ahmed, and E. Koh, "Attention models in graphs: a survey," ACM Trans. Knowl. Discov. Data **13**, 1–25 (2019).

[25] Z. Wu, P. Shirui, C. Fengwen, L. Guodong, Z. hengqi, and P. S. Yu, "A comprehensive survey on graph neural networks," IEEE Trans. Neural Netw. Learn. Syst. **32**, 4–24 (2020).

[26] J. Zhou, C. Ganqu, H. Shengding, Z. Zhengyan, Y. heng, L. Zhiyuan, W. Lifeng, L. Changcheng, and S. Maosong, "Graph neural networks: A review of methods and applications," AI Open **1**, 57–81 (2020).

[27] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," Proc. IEEE Signal Process. Mag. **34**, 18–42 (2017).

[28] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *Proc. ICLR* (Cornell, NY, USA, 2014), p. 6293.

[29] K. H. David, V. Pierre, and G. Rémi, "Wavelets on graphs via spectral graph theory," Appl. Comput. Harmon. Anal. **30**, 129–150 (2011).

[30] L. Yi, H. Su, X. Guo, and L. J. Guibas, "Syncspeccnn: Synchronize spectral cnn for 3D shape segmentation," *Proc. CVPR* (IEEE, Piscataway, NJ, 2017), pp. 6584–6592.

[31] J. Chen, T. Ma, and C. Xiao, "Fastgcn: fast learning with graph convolutional networks via importance sampling," *Proc. ICLR* (Cornell, NY, USA, 2018), p. 10247.
[32] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, "Generalized multidimensional scaling: a framework forisometry-invariant partial surface matching," *Proc. National Academy of Sciences (P-NAS)* (PNAS, Washington, DC, 2006), Vol. 103, pp. 1168–1172.
[33] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," *Proc. ICLR* (Banff, Canada, 2014).
[34] M. Belkin, J. Sun, and Y. Wang, "Discrete Laplace operator on meshed surfaces," *Proc. 24th ACM Symposium on Computational Geometry 773* (ACM, New York, NY, 2008), pp. 278–287.
[35] S. Bu, Z. Liu, J. Han, J. Wu, and R. Ji, "Learning high-level feature by deep belief networks for 3D model retrieval and recognition," IEEE Trans. Multimed **24**, 2154–2167 (2014).
[36] J. Xie, Y. Fang, and F. Zhu, "Deep Shape: Deep learned shape descriptor for 3D shape matching and retrieval," Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. 1275–1283 (2015).
[37] Z. Z. Han and Z. B. Liu, "BoSCC: bag of spatial context correlations for spatially enhanced 3D shape representation," IEEE Trans. Image Process. **26**, 3707–3720 (2017).
[38] X. H. Liu, Z. Z. Han, Y. S. Liu, and M. Zwicker, "Point2Sequence: learning the shape representation of 3D point clouds with an attention-based sequence to sequence network," *Proc. AAAI Conf. on Artificial Intelligence* (MIT Press, Cambridge, MA, 2019), pp. 8778–8785.
[39] S. Kumawat and S. Raman, "LP-3DCNN: unveiling local phase in 3D convolutional neural networks," *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2019), pp. 4903–4912.
[40] J. X. Liu, B. B. Ni, C. Y. Li, J. C. Yang, and Q. Tian, "Dynamic points agglomeration for hierarchical point sets learning," *Proc. IEEE Int'l. Conf. on Computer Vision (ICCV)* (IEEE, Piscataway, NJ, 2019), pp. 7546–7555.
[41] D. J. Zhang, F. Z. He, Z. G. Tu, L. Zou, and Y. L. Chen, "Pointwise geometric and semantic learning network on 3D point clouds," Integr. Comput.-Aided Eng. **27**, 57–75 (2020).
[42] W. K. Wu, Y. Zhang, D. Wang, and Y. Q. Lei, "SK-Net: Deep Learning on Point Cloud via End-to-end Discovery of Spatial Keypoints," *Proc. Thirty-Fourth AAAI Conf. on Artificial Intelligence* (MIT Press, Cambridge, MA, 2020), pp. 6422–6429.
[43] Q. Zheng, P. Zhao, Y. Li, H. Wang, Y. Yang, Q. Zheng, X. Tian, M. Yang, Y. Wu, and H. Su, "Spectrum interference-based two-level data augmentation method in deep learning for automatic modulation classification," Neural Comput. Appl. **33**, 7723–7745 (2020).
[44] Q. Zheng, X. Tian, M. Yang, Y. Wu, and H. Su, "PAC-Bayesian framework based drop-path method for 2D discriminative convolutional network pruning," Multidimens. Syst. Signal Process. **31**, 793–827 (2020).
[45] Q. Zheng, M. Yang, X. Tian, N. Jiang, and D. Wang, "A full stage data augmentation method in deep convolutional neural network for natural image classification," Discrete Dyn. Nature Soc. **2020**, 1–11 (2020).