Computer-Aided Design 160 (2023) 103512

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

SCMS-Net: Self-Supervised Clustering-Based 3D Meshes Segmentation Network

Xue Jiao [a,b], Yonggang Chen [a], Xiaohui Yang [b,*]

[a] School of Mathematics, Henan Institute of Science and Technology, Xinxiang, 453003, China
[b] Henan Engineering Research Center for Artificial Intelligence Theory and Algorithms, School of Mathematics and Statistics, Henan University, Kaifeng, 475000, China

ARTICLE INFO

Article history:
Received 27 December 2022
Accepted 5 March 2023

Keywords:
3D mesh segmentation
Unsupervised segmentation
Self-supervised learning
Deep clustering

ABSTRACT

The superior performance of deep learning in different domains has sparked significant interest in its applicability to 3D computer graphics. Deep learning has become the dominant technical architecture in current 3D mesh segmentation. However, learning-based 3D segmentation methods usually rely on high-quality training datasets, which are not readily available in practical applications. How to segment 3D meshes without exhaustive label annotations remains a challenging problem, especially in the context of deep learning. As a subset of unsupervised learning methods, self-supervised learning offers a promising learning paradigm for unlabeled 3D mesh segmentation. In this paper, we introduce a self-supervised clustering-Based network specifically for the segmentation of label-free 3D meshes. Our self-supervised clustering-based 3D mesh segmentation network (SCMS-Net) employs a two-branch architecture to learn effective feature representation. The two branches are unified into an end-to-end framework using a self-supervised strategy. Finally, the label predictions of the parts are generated by iterative clustering. We conducted ablation studies and comparative experiments on a standard benchmark to demonstrate the effectiveness of our approach.

1. Introduction

3D shape segmentation is an essential topic in computer vision and graphics, which aims to assign a category label to each part. As the basis and prerequisite of 3D shape modeling and understanding, 3D mesh segmentation has a wide range of applications, such as augmented reality, scene understanding, and shape generation [1–3]. Traditional segmentation methods focused on decomposing a single shape into meaningful parts by leveraging shape features. However, they are not robust enough to handle the structural variations present in various 3D shapes, which limits the ability of hand-crafted shape features to generalize in large-scale 3D shape segmentation [4,5].

In recent years, the remarkable achievements of deep learning in image processing tasks have sparked great interest in its application to 3D mesh segmentation. The availability of large scale labeled datasets is one essential requirement for the great success of deep learning in computer vision. Most supervised learning methods derived from sufficient well-annotated data can achieve promising segmentation results. However, the collection and annotation of 3D meshes are time-consuming and laborious. Moreover, humans hold different insights about the consistency of component annotation. These factors make it challenging to obtain well-annotated 3D datasets for deep neural network training. Unlike the significant progress in supervised segmentation methods, unsupervised segmentation methods for 3D meshes are still under-researched. How to segment 3D meshes without exhaustive label annotations remains a challenging problem, especially in the context of deep learning.

More recently, as a branch of unsupervised learning, self-supervised learning offers a promising learning paradigm for unlabeled 3D mesh segmentation. In self-supervised learning, models are learned by solving various pretext tasks, in which the supervision signals are automatically acquired from data. As a property-based pretext task in the self-supervised learning [6], deep clustering methods have attracted more attention. Motivated by the remarkable performance of deep clustering in unsupervised representation learning, we wondered whether it could effectively solve the segmentation task of label-free 3D mesh data. To the best of our knowledge, self-supervised clustering methods for 3D mesh segmentation remain unexplored. In this paper, we introduce a self-supervised clustering-based segmentation network (SCMS-Net) specifically for the segmentation of unlabeled 3D meshes.

In the absence of ground-truth labels, many deep clustering methods have achieved encouraging performances, such as DEC [7], DAC [8], IIC [9]. These methods have the advantage of

* Corresponding author.
E-mail address: xhyanghenu@163.com (X. Yang).

https://doi.org/10.1016/j.cad.2023.103512
0010-4485/

extracting useful representations from the data itself, but they rarely consider the structural information of the data during representation learning. However, for 3D mesh segmentation, both geometric features and structural informations are important for identifying different segments. Therefore, the deep clustering framework we use to implement the segmentation of 3D meshes should have the ability to integrate structural information. To arrive at this, we consider employing a latest work, structural deep clustering network (SDCN) [10], as the basic segmentation framework. SDCN uses the DNN, GCN, and dual self-supervised modules to fusion structure and geometric information.

Same with SDCN, our proposed end-to-end trainable mesh segmentation network consists of two branches. One branch employs a classical autoencoder network to learn the latent representations of geometric features. The other parallel branch is a graph autoencoder for capturing long-range structural informations between segmentation parts. Unlike SDCN, we developed an improved graph autoencoder branch that makes better use of the adjacency and face features of 3D mesh. Considering the presence of a large number of faces in the mesh, using fewer graph convolution layers, e.g., two to three layers, is insufficient to integrate structural information. Inverse, stacking more graph convolution layers will lead to over-smoothing problems. Our method introduces the skip connections in the encoder module to mitigate this drawback. In addition, we add the decoder module in the graph branch to reconstruct the structure information. The loss function of the whole segmentation network is composed of the reconstruction loss and the clustering loss. Different from SDCN, we use the reconstruction objectives of both branches to define the loss function to improve our segmentation network's capability.

An illustration of the overall framework is given in Fig. 1. Our deep clustering-based mesh segmentation network (SCMS-Net) employs a two-branch architecture to solve the unsupervised segmentation task. One branch is a classical autoencoder network for learning the effective representation of the feature space. The information is learned from autoencoder branch layers by considering the local cues. On the other side, considering the variations of structural information, we use an improved graph autoencoder as the parallel branch to capture long-range semantic relationships between parts. Then, we further integrate the geometric features and structure informations extracted from two branches to generate comprehensive representations. After that, the two-branch network is unified into an end-to-end framework using a self-supervised strategy. Finally, the label predictions of the parts are generated by iterative clustering.

Our method does not require training data with manual annotation, which reduces tedious and time-consuming annotation processing and is more practical than supervised learning-based segmentation methods. We have conducted extensive comparative experiments and ablation studies on the Princeton Segmentation Benchmark (PSB) dataset [11] and Shape COSEG dataset [12], both of which are publicly available datasets for 3D mesh segmentation. The experimental results show that our method can achieve comparable performance to existing segmentation methods.

The main contributions of our method can be summarized as follows:

- To the best of our knowledge, this is the first survey paper to explore the self-supervised clustering methods for the unsupervised segmentation task of 3D triangle meshes.
- In order to aggregate geometric features and structural information of 3D shapes, we develop a two-branch architecture based on a structural deep clustering network. Rather than using the graph convolutional network to reveal the underlying structure of data, we use an improved graph autoencoder branch to get the structure information of 3D meshes.
- The segmentation loss function consists of two components. One is a clustering loss defined by using a self-supervised mechanism, and the other is a reconstruction loss function designed to recover the geometric features and structural information of the two branches.

We conducted extensive experiments and detailed analysis on a benchmark dataset to validate the efficacy of the proposed network. In the rest of the paper, we first review related work in Section 2. Then, we describe the details of our SCMS-Net framework in Section 3. Next, Section 4 compares our approach with the current state-of-the-art methods and reports the experimental results on benchmark datasets. At last, the conclusions are presented in Section 5.

2. Related work

This section introduces the references related to our work. We provide an overview of deep clustering, followed by a brief introduction to mesh segmentation methods.

2.1. Deep clustering

Clustering is one of the most fundamental data analysis tasks. Recently, the breakthroughs in deep learning have led to a paradigm shift in artificial intelligence and machine learning, and the deep clustering has caught significant attention [13–18]. The basic idea of deep clustering is to integrate deep feature learning with the clustering objective. Deep clustering has been applied for self-supervised learning to relieve the burden of large-scale dataset annotation.

Many deep clustering methods have been proposed in the past few years, such as DEC [7] proposed a deep embedded clustering algorithm that learns feature representations and cluster assignments using deep neural networks. The clustering process of DEC consists of two main stages: one is the parameter initialization with a deep autoencoder, and the other is parameter optimization by iteratively computing between an auxiliary target distribution and minimizing the KL-divergence. DAC [8] adopts a binary pairwise-classification framework for image clustering. In DAC, the similarities are calculated as the cosine distance between label features of images which are generated by a deep convolutional network. IIC [9] is proposed to learn a clustering assignment by maximizing the mutual information between an image and its augmentations. DeepCluster method [19] iteratively between k-means clustering the features produced by the convent and updating the weights by predicting the cluster assignments as pseudo-labels. However, these deep clustering methods only focus on learning the representation from the data itself but ignore the structural information of the data during representation learning. Recently, SDCN [10] integrates the structural information into deep clustering by transferring the representations learned by the autoencoder to the corresponding GCN layer. However, these deep clustering methods are mainly used to handle image clustering or graph classification tasks and have not been applied to 3D mesh data. Inspired by SDCN, in this paper, we further explore whether similarity architectures can be applied to 3D mesh data to achieve the segmentation of unlabeled 3D meshes.

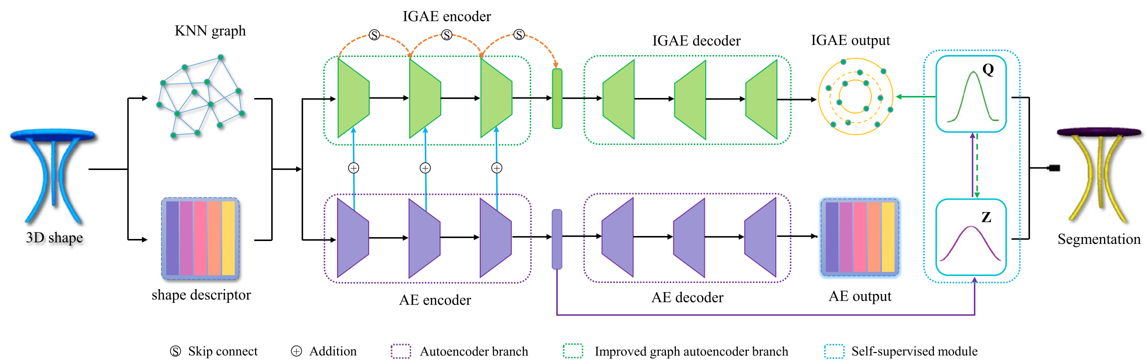

Fig. 1. Overview of our Self-supervised clustering-based mesh segmentation network (SCMS-Net). The proposed segmentation network is a two-branch architecture. One branch is a classical autoencoder model (AE), and the other is an improved graph autoencoder model (IGAE). Given a 3D mesh as input, we first preprocess the mesh to compute the shape descriptor and the KNN graph. Then, the descriptor and graph are entered into the corresponding branches separately. Next, we combine the graph structure and feature descriptor to generate robust semantic information. The two-branch network is unified using a self-supervised strategy to perform end-to-end clustering. Finally, the predicted label of parts depends on the clustering category to which it corresponds. Our method does not require manual annotation, so the prediction of the segmented labels is purely unsupervised.

2.2. Mesh segmentation

Mesh segmentation has long been studied in the field of computer vision and graphics. Early researches for mesh segmentation have a significant concern in finding practical features for part decomposition of single mesh [20–22]. However, the information in a single shape is insufficient to establish the correspondence between shape components. Due to the excellent neural network performance in image processing, various supervised methods perform 3D segmentation using deep neural networks trained on a large number of labeled 3D shapes or scenes [23–28].

Kalogerakis et al. [29] introduced a deep architecture for supervised learning, which maps the meshes to multiple 2D projections, and performs view-based inference by repurposing image-based deep networks. Liu et al. [30] explore the possibility of segmenting 3D models by taking advantage of realistic 2D images freely available online. Hao and Fang [31] formalize learning of a 3D shape segmentation function space as a meta-learning problem. To alleviate the dependency of a well-labeled dataset, Yuan and Fang [26] proposed a novel semi-supervised approach, named robust learning of one shot 3D Shape Segmentation, which only requires one single exemplar labeled shape for training. In [25], Shu et al. propose a weakly supervised segmentation method that uses sparse scribble-based labels instead of full high-quality labels to train the mesh segmentation model. Chen et al. [32] present a branched autoencoder for weakly supervised shape co-segmentation. Hu et al. [33] presented an unsupervised approach for co-segmentation by over-segmenting the input models into primitive patches and then grouping similar patches via a subspace clustering scheme. Wu et al. [34] suggested a spectral clustering method that generates consistent segmentation by performing spectral clustering in a fused space of shape descriptors. Shu et al. [35] first propose a novel unsupervised algorithm for automatically segmenting a single 3D shape or co-segmenting a family of 3D shapes using deep learning.

Most learning-based mesh segmentation methods achieve promising performance based on supervised and semi-supervised learning from a large volume of a well-annotated dataset. However, it is time-consuming and laborious to have a large volume of well-annotated shape segmentation datasets for training deep neural networks. In this work, we explore how to apply deep clustering to 3D mesh data. To alleviate the dependency on the well-labeled dataset, we propose a novel unsupervised mesh segmentation network based on the deep clustering method. Although deep clustering has been used in 3D shape processing with point clouds and voxel representations, to our best knowledge, we are the first to apply deep clustering to 3D shape segmentation with mesh-based representations.

3. Proposed method

3.1. Problem statement

Given a 3D shape depicted by mesh $M = \{V, F\}$, $V = \{v_i | v_i \in \mathbb{R}^3\}$ is set of mesh vertices, $F = \{f_i | f_i \in \{1, \dots, |V|\}^3\}$ is set of mesh faces. Our method aims to assign a pseudo label l_i to the face f_i so that mesh faces with similar semantic information are grouped into the same meaningful part. To resolve this problem, We follow the deep clustering paradigm that predicts pseudo-labels of mesh faces by iteratively clustering the semantic features and updating the parameters of the neural networks.

We first pre-compute shape descriptors X to describe the geometric features of the input mesh model; each row x_i of X is a feature descriptor of the triangular face f_i. Then, we consider x_i as the nodes v_i of the graph and construct an undirected graph to encode the shape structure information. We use a two-branch network based on the SDCN method to combine geometric and structural information to generate robust semantic features, and then unify the two-branch network into an end-to-end framework with the help of a self-supervised strategy. Finally, the label prediction of graph nodes is calculated by iteratively clustering, and the segmentation label l_i of the triangular face f_i depends on the final class of the graph node v_i to which it corresponds.

3.2. Shape descriptor extraction

Extracting shape descriptors based on geometric features is an efficient way to capture shape information. Considering that sophisticated feature computation will raise the time consumption of preprocessing, we use several essential features to describe the geometric information in the preprocessing step. Precisely, we extract a 114-dimensional feature vector for each mesh face containing: position (6-dim), average geodesic distance (3-dim), curvature (40-dim), shape diameter function (20-dim), and eigenvector features (45-dim).

The position feature consists of the face normal and centroid. The average geodesic distance (AGD) feature [36] of the face f_i is the vector composed of the average geodesic distance of the three vertices. We employ the multiscale fitting algorithm [37] to estimate vertex's curvature feature: k_1, k_2, $(k_1+k_2)/2$, k_1k_2, d_1, d_2. Then, the curvature features of f_i are composed of the curvature features of three vertices concatenated with the average values.

k_1 and k_2 are the principal curvature values, d_1 and d_2 are the principal curvature directions. This results in a 40-dimension curvature feature. Same with [38], the SDF features [39] are computed using cones of angles 30, 60, 90, and 120. For each cone, we get the weighted average of the samples with different normalizing parameters $\alpha = 1, \alpha = 2, \alpha = 4, \alpha = 8$. This yields 20 features representing different approximations of the local shape diameter. We compute the Laplace–Beltrami matrix on the mesh and generate the eigenvalues and eigenvectors by solving the generalized eigenvalue problem using ARPACK [40]. We retain each mesh vertex's corresponding eigenvectors of the first 15 largest eigenvalues. The eigenvectors of face f_i are computed by concatenating the eigenvectors of three vertices. This yields a 45-dimension feature.

3.3. Graph construction

Given a triangle mesh M, we generate the shape descriptor $X = \{x_1, x_2, \ldots, x_N\} \in \mathbb{R}^{N\times D}$ after feature precomputation, where N is the number of mesh faces, D is the dimension of the shape descriptor. We use an undirected graph to encode the topological structure of the input mesh. We treat x_i as a graph node and establish graph edges according to their feature similarity.

We employ the K-nearest neighbors (KNN) method [41] to select the top-k correlation neighbors for each node. The correlation similarity of two graph nodes is calculated by the Gaussian kernel function:

$$S_{ij} = e^{\frac{-\|x_i - x_j\|^2}{\tau}} \tag{1}$$

where τ is the parameter in the heat conduction equation, we set the value of τ in our experiments to 0.5. For each graph node v_i, a graph edge e_{ij} is added to connect it with its neighbor v_j. Based on this, we can generate the adjacency matrix A, which is a binary matrix with the term $A_{ij} = 0$ if $e_{ij} \in E$, and $A_{ij} = 1$ if $e_{ij} \notin E$.

As defined above, we represent the input mesh M to a KNN graph, which can be characterized by the similarity matrix $S = [S_{ij}]$ and adjacency matrix $A = [A_{ij}]$. We set the value of k to 10 to take into account the trade-off between computational efficiency and segmentation performance.

3.4. SCMS-Net

This section presents a detailed description of our proposed SCMS-Net. We apply a two-branch structure that combines geometric feature and shape structure to predict part labels better. One branch is an autoencoder branch for learning the effective data representation of the feature space. The other is an improved graph autoencoder branch for capturing long-range semantic relationships between graph nodes. The input of the segmentation network is the feature graph X with the adjacency matrix A, and the output is the prediction labels for mesh faces. By utilizing a self-supervised strategy, we can handle the mesh segmentation task in an unsupervised manner.

We use the classical autoencoder to learn the geometric representation of different kinds of mesh models. This autoencoder consists of two parts: an encoder module and a decoder module. The encoder module combines fully connected layers to map the input geometric shape tensor to a low-dimensional latent feature space. The decoder module reconstructs the input features through several fully connected layers. In our approach, the encoder and decoder modules have three layers each, and the input of this branch is the shape descriptor X of the input mesh with ReLU applied as the activation function. We use $\widehat{H}$ to denote the output of the decoder part of the autoencoder branches.

The autoencoder branch learns valuable shape representations from mesh model but disregards the relationships between shape parts. To this end, the SDCN method [10] uses multiple graph convolution layers to learn structural information and then takes a transfer manner to fuse structure and data representation. Inspired by SDCN, we adopt a similar approach to process the structural information of the meshes. However, unlike SDCN, which uses graph convolution layers to form structural branches, we use a modified graph autoencoding architecture as the structural branch to get potential structure representations.

Specifically, the encoder part of the graph autoencoder branch consists of three graph convolution layers. We combine the structural information generated by the graph convolution layer and the geometric representation learned by the autoencoder branch to produce a comprehensive representation, the equation can be expressed by

$$\widetilde{G}^{(l)} = \lambda G^{(l)} + (1-\lambda)\, H^{(l)} \tag{2}$$

where $G^{(l)}$ is the structural information of the lth graph convolution layer, and $H^{(l)}$ is the geometric representation of the corresponding layer of autoencoder branch. λ is a balance coefficient, we uniformly set it to 0.5 in the experiments.

In experiments, we found that the three graphical convolution layers cannot adequately represent the structural information of the mesh model due to the presence of tens of thousands of faces in the mesh model. We tried to increase the number of graphical convolution layers, but this led to the problem of over-smoothing of graphical convolution. As suggested by [42,43], we introduce skip connection in the encoder module to obtain robust latent representations. In our method, the layer in the encoder part of the graph autoencoder branch is formulated as.

$$\widetilde{G}^{(l+1)} = \mathcal{F}\left(\widetilde{G}^{(l)}, \widetilde{A}\right) + h\left(\widetilde{G}^{(l)}\right) \tag{3}$$

where $\mathcal{F}$ is the general graph convolution operation, $\widetilde{A}$ is the normalized adjacent matrix, h is the fully connection mapping to facilitate the vertex-wise addition of layers of different dimensions.

Symmetrically with the encoder module of graph autoencoder, the decoder consists of three layers. The encoder part reconstructs the structural information matrix and the adjacency matrix. As the initial kNN graph construction, we use the correlation similarity of graph nodes to reconstruct the affinity matrix. We use $\widehat{G}$ to denote the output structure information of the decoder module and $\widehat{A}$ to denote the reconstructed adjacency matrix. The extensive experiments on the mesh segmentation task verify the impact of these ideas on various shape classes.

3.5. Self-supervised strategy

Now, we have associated autoencoder with graph autoencoder in mesh segmentation network architectures. However, neither of them can be immediately applied to deep clustering problems. We apply the same self-supervised module as SDCN to unify the branches of autoencoder and graph autoencoder into a unified framework. For the autoencoder branch, we calculate the similarity between ith sample h_i and jth clustering center u_j by using the Student's t-distribution as the kernel:

$$z_{ij} = \frac{\left(1 + \|h_i - u_j\|^2/v\right)^{-\frac{v+1}{2}}}{\sum_{j'}\left(1 + \|h_i - u_{j'}\|^2/v\right)^{-\frac{v+1}{2}}} \tag{4}$$

$$q_{ij} = \frac{z_{ij}^2 / \sum_i z_{ij}}{\sum_{j'}\left(z_{ij'}^2 / \sum_i z_{ij'}\right)} \tag{5}$$

In Eq. (7), h_i is the ith row of $\widehat{H}$, K-means initialize u_j on representations learned by the pre-train autoencoder, and v are the degrees of freedom of the Student's t-distribution. z_{ij} indicates the soft assignment probability of the ith node to the jth center. The soft assignment probability of the ith node to the jth center is represented by z_{ij}. In order to increase the confidence in cluster assignment, we introduce Eq. (8) to drive all samples closer to cluster centers. $Q = [q_{ij}]$, $0 \leq q_{ij} \leq 1$ is an element of the generated target distribution, which indicates the probability of the ith sample belongs to the jth cluster center.

The clustering loss function of the autoencoder is defined by adapting the KL-divergence in the following form:

$$L_{C_1} = \sum_i \sum_j q_{ij} log \frac{q_{ij}}{z_{ij}} \tag{6}$$

By minimizing this KL-divergence loss, the target distributions P and Q are iteratively updated by supervising each other. This self-supervised mechanism assists the AE module in learning a better representation for the clustering task.

For the graph convolution branch, we adopt a similar strategy. We first calculate soft target distributions for the clustering embedding $\widetilde{G}$ which has integrated the information from both autoencoder and improved graph autoencoder. $\widetilde{G}$ is the encode embedding layer of graph autoencoder with the softmax function. We consider $\widetilde{G}$ as a probability distribution. Then, we use distribution Q to supervise distribution $\widetilde{G}$ as follows:

$$L_{C_2} = \sum_i \sum_j q_{ij} log \frac{q_{ij}}{\widetilde{g}_{ij}} \tag{7}$$

where $\widetilde{g}_{ij} \in \widetilde{G}$ indicates the probability sample i belongs to cluster center j.

3.6. Network loss and training

In our method, the overall segmentation network loss function consists of two parts. One is the clustering loss, and the other is the reconstruction loss of two branches. Same with SDCN, to train the network in a unified framework and improve the representative capability of each branch, the clustering loss L_C is defined by using the self-supervised mechanism:

$$L_C = L_{C_1} + L_{C_2} \tag{8}$$

Unlike SDCN defines the reconstruction loss functions only for the autoencoder branch, our reconstruction loss function is designed for both two branches. The reconstruction loss of the autoencoder branch is given by

$$L_{AE} = \frac{1}{2N} \sum_{i=1}^{N} \|X - \widehat{H}\|_F^2 \tag{9}$$

For the graph autoencoder branch, we define the reconstruction loss of structural information based on the adjacency matrix and feature matrix of the graph nodes.

$$L_{IGAE} = \frac{1}{2N} \sum_{i=1}^{N} \|AX - \widehat{A}\widehat{G}\|_F^2 \tag{10}$$

where $\widehat{H}$ to denote the output of the decoder part of the autoencoder branches, $\widehat{G}$ is the output structure information of the decoder module, and $\widehat{A}$ is the reconstructed adjacency matrix, N is the number of graph nodes. Then the reconstruction loss function becomes $L_R = L_{AE} + L_{IGAE}$. The whole segmentation loss L_S can be formulated as follows:

$$L_S = \alpha L_R + \beta L_C \tag{11}$$

where α and β are pre-defined hyperparameters that balance the importance of reconstruction and clustering.

In the segmentation loss formulation, KL divergence gently updates the entire segmentation model to avoid significant disturbances in the data representation. Two branches are unified in the same optimization objective, making their results converge during the training process. The detailed training process of the proposed segmentation network is shown in Algorithm 1.

Algorithm 1 Training procedure of SCMS-Net.

Input: Shape descriptor X; Adjacency matrix A; Cluster number K; Iteration number I; Hyper-parameters α and β.
Output: Segmentation labels of mesh faces.

1: Initialize the network parameters of two branches with uniform initialization method.
2: Initialized clustering centers u_j with K-means for potential representations in the pre-trained autoencoder branch.
3: **for** $i = 1$ to I **do**
4: Calculate latent representations of two branches;
5: Transfer the representations learned from $H^{(l)}$ into a corresponding graph encoder layer by Eq. (2);
6: Generate the next layer representation of the graph convolution by Eq. (3);
7: Compute loss function L_C;
8: Compute the loss functions L_R;
9: Update the whole network by minimizing Eq. (11).
10: **end for**
11: Generate the prediction labels of graph nodes with the final output of encoder module of $\widehat{G}$ by K-means.
12: Get the segmentation label of mesh face f_i according to the final cluster of the corresponding graph node.
13: **return** Segmentation labels of mesh faces.

4. Experiments

4.1. Datasets

We test our unsupervised segmentation model on the Princeton Segmentation Benchmark (PSB) dataset [11], Shape COSEG dataset [12], and the Human Body dataset [44]. The PSB dataset contains 380 manually-labeled segmentation meshes in 19 categories, and each category contains 20 meshes with different shape configurations. All models in the PSB have been carefully re-meshed and reconstructed such that their mesh representation is watertight with clean topology. The Human Body dataset consists of 399 meshes from SCAPE [45], FAUST [46], SHREC07 [47], and the meshes are manually segmented into eight parts. The COSEG dataset contains 190 segmented models in 8 small categories, plus 900 synthetic variations of those in 3 large categories. We evaluate the performance of our approach on the three largest sets: Vases (300 meshes), Chairs (400 meshes), and Tele-aliens (200 meshes).

4.2. Evaluation metrics

We use the mean accuracy (mAcc), overall accuracy (oAcc), and mean intersection over union (mIoU) as the evaluation metrics. The mAcc is the mean of each accuracy per segmentation part, and the oAcc is calculated as the total number of correctly classified mesh faces divided by the total number of mesh faces. The mIoU is the mean of IoU calculated by the ratio of intersection over the union of each segmentation part. We calculate the evaluation metrics for each shape and then average the evaluation metrics for all shapes included in each category to obtain the evaluation results. Since our method belongs to the category of unsupervised segmentation, ground truth labels are only used to evaluate the quality of the segmentation results, and they are not involved in our training procedure.

Table 1
Quantitative results by our method for each category of PSB dataset. For all three metrics, higher values indicate better performance.

Category	mIoU	mAcc	oAcc
Airplane	0.5839	0.6703	0.6779
Ant	0.5183	0.6595	0.6627
Bird	0.5346	0.6219	0.6405
Bearing	0.6057	0.6521	0.7998
Bust	0.3668	0.4291	0.4449
Chair	0.6208	0.7635	0.8279
Fish	0.5523	0.6437	0.6882
Fourleg	0.4604	0.5529	0.6091
Cup	0.6724	0.8496	0.8789
Glasses	0.6204	0.7666	0.8897
Hand	0.5499	0.6772	0.6854
Human	0.4316	0.5636	0.5801
Plier	0.4612	0.6578	0.6972
Octopus	0.9556	0.9788	0.9799
Table	0.9591	0.9796	0.9855
Teddy	0.4652	0.5957	0.6343
Mech	0.6538	0.8424	0.8645
Armadillo	0.3628	0.4275	0.4658
Vase	0.6473	0.7343	0.7594

4.3. Network configuration

Our method is implemented with the PyTorch platform and an NVIDIA RTX 2080Ti GPU. We pre-train the autoencoder independently for each mesh model with 50 epochs. Then, we train the whole network, which integrates both branches for 200 epochs. Adam optimizer [48] with an initial learning rate of 0.01 and stepwise rate decay schedule of 0.99 by ten steps is applied for network optimization. The dimension of hidden layers in the autoencoder branch is 112, 112, 56, 14, and the dimension of the input layer is 114. The dimension of each layer in the graph autoencoder branch is consistent with the autoencoder. We use LeakyReLU as an activation function and implement batch normalization [49] for the layers in the graph autoencoder. The hyperparameters α and β in Eq. (11) are set to 0.5 and 0.01 for all mesh models.

4.4. Timings

We also estimate the time efficiency of the model. Here we take a 3D mesh model with 15K as an example, the precomputation of shape descriptor takes about 85 s, and the most time-consuming procedure are the calculation of SDF, which take about 70% of the total extraction time; The *KNN* graph construction for the mesh took about 13 s. The training of segmentation network takes about 48 s for 200 epochs.

4.5. Segmentation performances

Since the unsupervised method is not guaranteed in advance to generate the same number of partitions as ground-truth. For comparison, we manually count the number of segmentation components for each model and assign it to the parameters of K-means to generate the same number of segments as ground-truth. The quantitative results per category of the PSB dataset evaluated are provided in Table 1.

As shown in Table 1, there are seven categories that both of mAcc and oAcc values are above 70%, and six categories in them have mIoU values above 60%. For example, our proposed method performs well for the categories of Bearing, Octopus, and Table. The categories with low evaluation values are Bust, Human, Armadillo, and Fourleg. The main reason is that these models have a great diversity of shapes and different gestures. Segmentation network difficult to capture discriminative shape and structure information.

Table 2
The results of ablation experiments on the input features. no-Position indicates the lack of positional features, the rest of these cases are similar.

Features	mIoU	mAcc	oAcc
no-Position	0.5164	0.6408	0.6946
no-Curvature	0.5238	0.6511	0.7027
no-SDF	0.5328	0.6768	0.6799
no-AGD	0.3799	0.5130	0.5433
no-Eigenvectors	0.3226	0.5211	0.5550
Ours	**0.5803**	**0.7053**	**0.7230**

4.6. Ablation studies

We perform three ablation studies (on the input geometric features, the loss function, and the two branches of the network) to justify some of the design and parameter choices in our architecture.

Input features. Feature extraction is a necessary part of the network model. Our segmentation network requires different features to represent the local and global geometry of the triangular faces of the mesh, respectively. For this purpose, we analyze the ability of network segmentation with different combinations of features. We evaluate the segmentation method using the average evaluation metrics for all categories of the PSB dataset. The quantitative results are given in Table 2. The feature vectors and AGD feature vectors are critical initial features that have the most significant impact on the accuracy of the results. The reason is that they represent the shape information and implicitly encode the connectivity information of the shape.

Branches. Our segmentation network consists of Autoencoder and Graph Autoencoder as the basic constituent structures. In this part, we illustrate the effectiveness of each branch. We compare our proposed network with the network using only autoencoder branch, the network using improved graph autoencoder branch, and SDCN network with two branches. We abbreviate them as AE-Branch, IG-Branch, and A&G-Branch, respectively. Among them, AE-Branch and IG-Branch uses L_{AE} and L_{IGAE} as the loss function, A&G-Branch still use L_S as the loss function. We perform experiments on the PSB dataset and the COSEG dataset. The segmentation performance are summarized in Fig. 2. Based on the results, our proposed network shows superior performance against the compared methods in most circumstances. S_{AE} merely exploit node representations for clustering, it seldom take structure information into account, leading to suboptimal performance. S_{GAE} and $S_{A\&G}$ under-utilize abundant information from data itself and might be limited to the over-smoothing phenomenon. our method incorporates geometric representations learned by AE into the whole clustering framework, and mutually explores structure information and geometric representations with a fusion module for consensus representation learning.

Loss function. The loss function of the whole segmentation network is composed of the reconstruction loss and the clustering loss. We evaluate the segmentation method using the average of the evaluation metrics for all categories of PSB dataset. The experimental results on the PSB dataset and the Human Body dataset are shown in Tables 3 and 4. L_R and L_C represent the reconstruction and clustering loss function, respectively. L_S is our designed segmentation loss function as defined in Eq. (11). In our experiments we found that for most categories, the reconstruction loss has a significant impact on the segmentation results, due to the fact that this part contains the main information of the mesh model. As described in Section 3, the clustering loss function defined based on the self-supervised mechanism unifies the branches of the autoencoder and the graph autoencoder so that their results converge during the training process. For

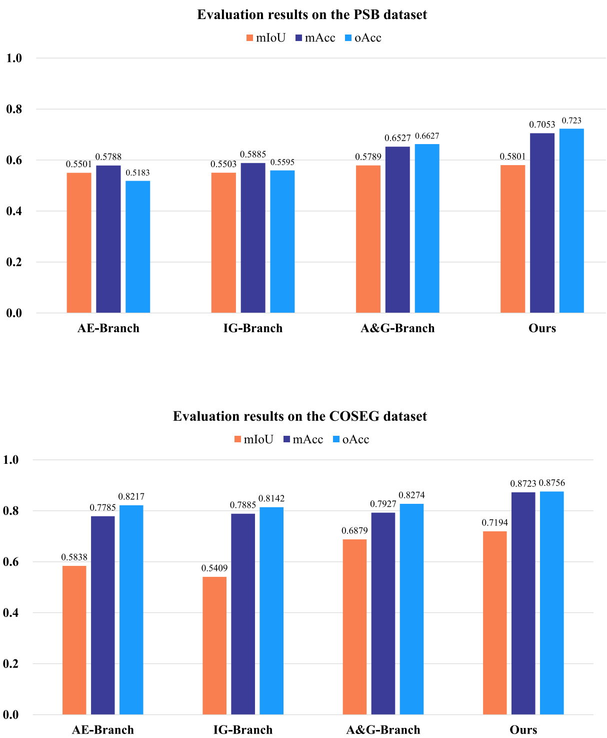

Fig. 2. The results of ablation experiments on different branches. AE-Branch represents our proposed network using only autoencoder branch, IG-Branch represents the proposed network using only improved graph autoencoder branch, A&G-Branch represents the SDCN network.

Table 3
The results of ablation experiments of the loss function on PSB dataset.

Loss function	mIoU	mAcc	oAcc
L_R	0.5364	0.6782	0.7095
L_C	0.3904	0.5755	0.5915
L_S	**0.5803**	**0.7053**	**0.7230**

Table 4
The results of ablation experiments of the loss function on the Human Body dataset.

Loss function	mIoU	mAcc	oAcc
L_R	0.4525	0.5123	0.5543
L_C	0.3862	0.4751	0.5024
L_S	**0.4934**	**0.5712**	**0.6401**

diverse segmentation classes, the clustering loss function can help improve the accuracy of segmentation in terms of average performance.

4.7. Comparison with the state-of-the-arts

We compare our segmentation method with the state-of-the-art segmentation methods on the COSEG dataset. Except for the classical unsupervised method, we also compare our method with several 3D shape segmentation methods, which employ fully supervised deep learning techniques to segment shapes. All the parameters are consistent with the experiments on the PSB dataset. We use the segmentation accuracy per category as the metric. Following [29], the accuracy is computed as the percentage of the area of correctly classified faces over the area of all the surfaces.

Table 5
The comparison results on the Human Body dataset.

Method	Accuracy
MeshCNN [50]	0.8530
PD-MeshNet [51]	0.8561
HodgeNet [52]	0.8503
Ours	**0.8582**

Table 6
The comparison results of our method with other segmentation methods. The performance of existing methods is based on publicly reported results in the literatures [53–55], and "-" means the result was not reported.

Method	Type	Vase	Chair	Alien
MeshCNN [50]	Supervised	0.9236	0.9299	**0.9626**
HodgeNet [52]	Supervised	0.9030	0.9568	0.9603
Laplacian2Mesh [53]	Supervised	0.9458	0.9658	0.9501
ShapePFCN [29]	Supervised	0.9060	0.9110	0.9570
ShapeBoost [38]	Supervised	0.8970	0.7650	0.8080
Guo et al. [56]	Supervised	0.8010	0.8080	0.8000
SFCN [54]	Supervised	**0.9591**	0.9117	0.9300
LaplacianNet [57]	Supervised	0.9420	0.9220	0.9390
DCN [55]	Supervised	0.9090	0.9570	–
Label Proposed	Supervised	0.9348	**0.9677**	0.9614
Wu et al. [34]	Un-supervised	0.8750	0.9060	0.7530
Sidi et al. [58]	Un-supervised	0.6990	0.8020	–
Proposed	Un-supervised	0.8744	**0.9183**	**0.8215**

Tables 5 and 6 show the segmentation accuracy per category of the Human Body dataset and the COSEG dataset. The best results are marked in bold. In the supervised segmentation setting, denoted by "label proposed", we generate initial pseudo-labels using our method, and use cross-entropy as the loss function for training. Table 5 shows the experiment performance of the proposed method on the Human Body dataset in supervised setting, and the experimental results show that our method achieves better segmentation results on this dataset. In Table 6, the performance of existing methods is based on publicly reported results in the literature, and "-" means the result was not reported. Experimental results show that our unsupervised segmentation method can achieve better-unsupervised segmentation results for both Chair and Alien categories. The supervised experimental results show that our method can obtain competitive performance compared to Laplacian2Mesh and HodgeNet.

4.8. Visualized examples

In this part of the experiment, we show the visualization results of our method on the PSB dataset and the shape COSEG dataset. Fig. 3 shows some samples of our segmentation results on the PSB dataset. Figs. 4–6 show some segmentation results on the Shape COSEG dataset. The experimental results show that our method performs well on most of the shape sets under the condition that there is no training data with annotations, and can achieve robust performance on various types of models with different datasets. However, the edges between different parts are not smooth, and our method has difficulty correctly classifying their nearby locations.

5. Conclusion

In this paper, we focus on the problem of unsupervised segmentation of a 3D model represented by a triangular mesh. We use the shape descriptor and undirected graph to describe the source mesh's local geometric and shape structure information. We developed a two-branch unsupervised segmentation

Fig. 3. Segmentation results on the PSB dataset by our method.

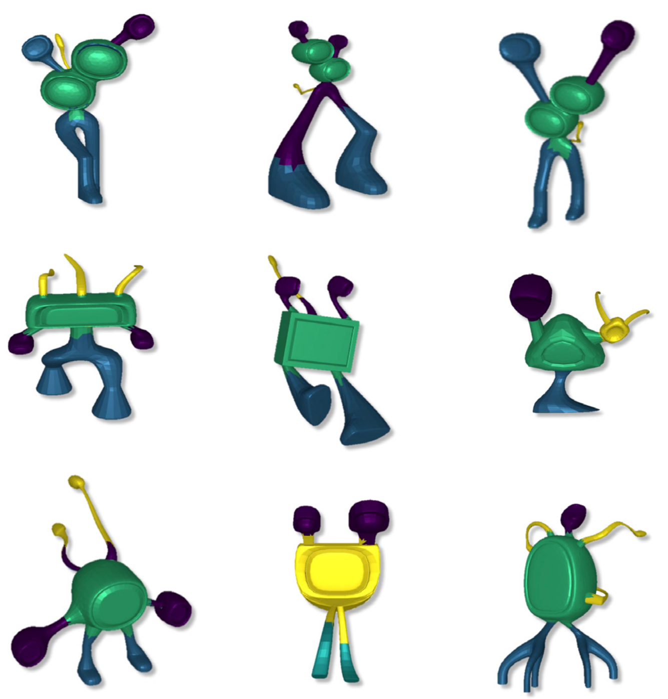

Fig. 4. Segmentation results on the Shape COSEG dataset (Aliens category) by our method.

framework that accomplishes label prediction by employing a self-supervised strategy. In contrast to supervised learning, our approach does not involve annotation training datasets. We conduct extensive experiments on the benchmark datasets to evaluate our method. The experimental results demonstrate that our

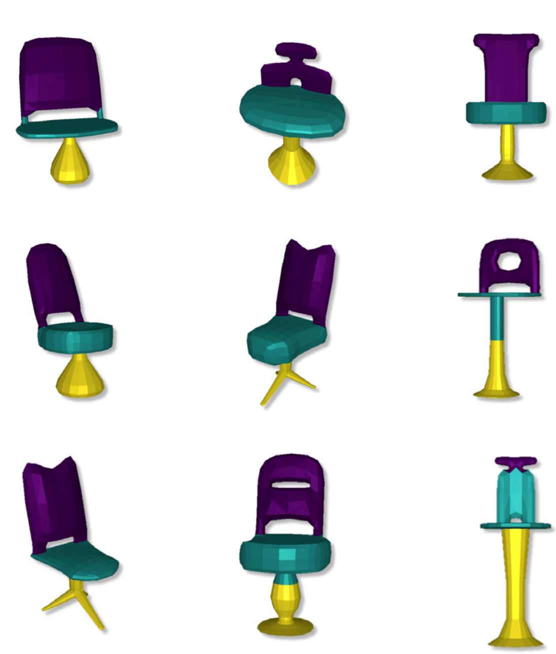

Fig. 5. Segmentation results on the Shape COSEG dataset (Chairs category) by our method.

Fig. 6. Segmentation results on the Shape COSEG dataset (Vases category) by our method.

method can achieve comparable performance. For the 3D models with complex topology and complicated properties, such as disconnected components, non-manifold edges, and self-intersections, our method cannot be directly used to deal with such meshes. We will consider how to handle such meshes in our subsequent work. Another possibility for future work is to investigate the mesh representation to hold for multi-resolution structures. Finally, we would like to consider extending our method to handle other related areas, such as point cloud segmentation, shape retrieval and other exciting directions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the Key Scientific Research Project in Colleges and Universities of Henan Province (22B110004), Key Scientific and Technological Research Projects in Henan Province (222102110409), and Henan Provincial Natural Science Foundation (232300420427).

References

[1] Mo K, Guerrero P, Yi L, Su H, Wonka P, Mitra NJ, et al. StructureNet. ACM Trans Graph 2019;38:1–19.
[2] Wang L, Li X, Fang Y. Few-shot learning of part-specific probability space for 3D shape segmentation. In: 2020 IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 4503–12.
[3] Yi L, Su H, Guo X, Guibas L. SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation. In: 2017 IEEE conference on computer vision and pattern recognition. 2017, p. 6584–92.
[4] Rodrigues RSV, Morgado JFM, Gomes AJP. Part-based mesh segmentation: A survey. Comput Graph Forum 2018;37:235–74.
[5] Shamir A. A survey on mesh segmentation techniques. Comput Graph Forum 2008;27:1539–56.
[6] Jing L, Tian Y. Self-supervised visual feature learning with deep neural networks: A survey. IEEE Trans Pattern Anal Mach Intell 2019;43:4037–58.
[7] Xie J, Girshick RB, Farhadi A. Unsupervised deep embedding for clustering analysis. 2015, arXiv abs/1511.06335.
[8] Chang J, Wang L, Meng G, Xiang S, Pan C. Deep adaptive image clustering. In: 2017 IEEE international conference on computer vision. 2017, p. 5880–8.
[9] Ji X, Vedaldi A, Henriques JF. Invariant information clustering for unsupervised image classification and segmentation. In: 2019 IEEE/CVF international conference on computer vision. 2018, p. 9864–73.
[10] Bo D, Wang X, Shi C, Zhu M, Lu E, Cui P. Structural deep clustering network. In: Proceedings of the web conference 2020. New York, NY, USA: Association for Computing Machinery; 2020, p. 1400–10.
[11] Chen X, Golovinskiy A, Funkhouser T. A benchmark for 3D mesh segmentation. In: ACM SIGGRAPH 2009 papers. New York, NY, USA: Association for Computing Machinery; 2009, p. 1–12.
[12] Wang Y, Asafi S, van Kaick OM, Zhang H, Cohen-Or D, Chen B. Active co-analysis of a set of shapes. ACM Trans Graph 2012;31:1–10.
[13] Hatcher WG, Yu W. A survey of deep learning: Platforms, applications and emerging research trends. IEEE Access 2018;6:24411–32.
[14] Jabi M, Pedersoli M, Mitiche A, Ayed IB. Deep clustering: On the link between discriminative models and K-means. IEEE Trans Pattern Anal Mach Intell 2018;43:1887–96.
[15] Shiran G, Weinshall D. Multi-modal deep clustering: Unsupervised partitioning of images. In: 2020 25th international conference on pattern recognition. 2019, p. 4728–35.
[16] Zhan X, Xie J, Liu Z, Ong YS, Loy CC. Online deep clustering for unsupervised representation learning. In: 2020 IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 6687–96.
[17] Huang Z, Chen J, Zhang J, Shan H. Exploring non-contrastive representation learning for deep clustering. 2021, arXiv abs/2111.11821.
[18] Dang Z, Deng C, Yang X, Wei K-J, Huang H. Nearest neighbor matching for deep clustering. In: 2021 IEEE/CVF conference on computer vision and pattern recognition. 2021, p. 13688–97.
[19] Caron M, Bojanowski P, Joulin A, Douze M. Deep clustering for unsupervised learning of visual features. In: Computer vision - ECCV 2018 - 15th European conference. Lecture notes in computer science, vol. 11218, Springer; 2018, p. 139–56.
[20] Huang Q, Wicke M, Adams B, Guibas LJ. Shape decomposition using modal analysis. Comput Graph Forum 2009;28:407–16.
[21] Zhang J, Zheng J, Wu C, Cai J. Variational mesh decomposition. ACM Trans Graph 2012;31:1–14.
[22] Au OK-C, Zheng Y, Chen M, Xu P, Tai C-L. Mesh segmentation with concavity-aware fields. IEEE Trans Vis Comput Graphics 2012;18:1125–34.
[23] Qian S, Liu L, Xu W, Lu C. iSeg3D: An interactive 3D shape segmentation tool. 2021, arXiv abs/2112.12988.
[24] Lin C, Liu L, Li C, Kobbelt LP, Wang B, Xin S, et al. SEG-MAT: 3D shape segmentation using medial axis transform. IEEE Trans Vis Comput Graphics 2020;28:2430–44.
[25] Shu Z, Shen X, Xin S, Chang Q, Feng J, Kavan L, et al. Scribble-based 3D shape segmentation via weakly-supervised learning. IEEE Trans Vis Comput Graphics 2020;26:2671–82.
[26] Yuan S, Fang Y. ROSS: Robust learning of one-shot 3D shape segmentation. In: 2020 IEEE winter conference on applications of computer vision. 2020, p. 1950–8.
[27] Sun C-Y, Yang Y-Q, Guo H, Wang P-S, Tong X, Liu Y, et al. Semi-supervised 3D shape segmentation with multilevel consistency and part substitution. 2022, arXiv abs/2204.08824.
[28] Chen S, Fang J, Zhang Q, Liu W, Wang X. Hierarchical aggregation for 3D instance segmentation. In: 2021 IEEE/CVF international conference on computer vision. 2021, p. 15447–56.
[29] Kalogerakis E, Averkiou M, Maji S, Chaudhuri S. 3D shape segmentation with projective convolutional networks. In: 2017 IEEE conference on computer vision and pattern recognition. 2017, p. 6630–9.
[30] Liu J, Rosin PL, Sun X, Xiao J, Lian Z. Image-driven unsupervised 3D model co-segmentation. Vis Comput 2019;35:909–20.
[31] Hao Y, Fang Y. Meta-learning 3D shape segmentation functions. 2021, arXiv abs/2110.03854.
[32] Chen Z, Yin K, Fisher M, Chaudhuri S, Zhang H. BAE-NET: Branched autoencoder for shape co-segmentation. In: 2019 IEEE/CVF international conference on computer vision. 2019, p. 8489–98.
[33] Hu R, Fan L, Liu L. Co-segmentation of 3D shapes via subspace clustering. Comput Graph Forum 2012;31.
[34] Wu Z, Wang Y, Shou R, Chen B, Liu X. Unsupervised co-segmentation of 3D shapes via affinity aggregation spectral clustering. Comput Graph 2013;37:628–37.
[35] Shu Z, Qi C, Xin S, Hu C, Wang L, Zhang YG, et al. Unsupervised 3D shape segmentation and co-segmentation via deep learning. Comput Aided Geom Design 2016;43:39–52.
[36] Hilaga M, Shinagawa Y, Kohmura T, Kunii TL. Topology matching for fully automatic similarity estimation of 3D shapes. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques. New York, NY, USA: Association for Computing Machinery; 2001, p. 203–12.
[37] Meyer M, Desbrun M, Schröder P, Barr AH. Discrete differential-geometry operators for triangulated 2-manifolds. In: Visualization and mathematics III. Berlin, Heidelberg: Springer Berlin Heidelberg; 2003, p. 35–57.
[38] Kalogerakis E, Hertzmann A, Singh K. Learning 3D mesh segmentation and labeling, vol. 29. 2010, p. 1–12.
[39] Shapira L, Shalom S, Shamir A, Cohen-Or D, Zhang H. Contextual part analogies in 3D objects. Int J Comput Vis 2010;89:309–26.
[40] Maschhoff KJ, Sorensen DC. A portable implementation of ARPACK for distributed memory parallel architectures. 1996.
[41] Ramaswamy S, Rastogi R, Shim K. Efficient algorithms for mining outliers from large data sets. SIGMOD Rec 2000;29(2):427–38.
[42] Li G, Müller M, Thabet AK, Ghanem B. DeepGCNs: Can GCNs go as deep as CNNs? In: 2019 IEEE/CVF international conference on computer vision. 2019, p. 9266–75.
[43] Chen T, Zhou K, Duan K, Zheng W, Wang P, Hu X, et al. Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study. 2022, arXiv aabs/2108.10521.
[44] Maron H, Galun M, Aigerman N, Trope M, Dym N, Yumer E, et al. Convolutional neural networks on surfaces via seamless toric covers. ACM Trans Graph 2017;36:1–10.
[45] Anguelov D, Srinivasan P, Koller D, Thrun S, Rodgers J, Davis J. SCAPE: shape completion and animation of people. ACM Trans Graph 2005;24:408–16.
[46] Bogo F, Romero J, Loper M, Black MJ. FAUST: Dataset and evaluation for 3D mesh registration. In: 2014 IEEE conference on computer vision and pattern recognition. 2014, p. 3794–801.
[47] Giorgi D, Biasotti S, Paraboschi L. SHape REtrieval Contest 2007: Watertight models track. 2007.
[48] Kingma DP, Ba J. Adam: A method for stochastic optimization. 2015, arXiv abs/1412.6980.
[49] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd international conference on machine learning. JMLR workshop and conference proceedings, vol. 37, 2015, p. 448–56.

[50] Hanocka R, Hertz A, Fish N, Giryes R, Fleishman S, Cohen-Or D. MeshCNN: a network with an edge. ACM Trans Graph 2019;38:1–12.
[51] Milano F, Loquercio A, Rosinol A, Scaramuzza D, Carlone L. Primal-dual mesh convolutional neural networks. 2020, arXiv abs/2010.12455.
[52] Smirnov D, Solomon JM. HodgeNet: Learning spectral geometry on triangle meshes. ACM Trans Graph 2021;40:1–11.
[53] Dong Q, Wang Z, Gao J, Chen S, Shu Z, Xin S. Laplacian2Mesh: Laplacian-based mesh understanding. 2022, arXiv abs/2202.00307.
[54] Wang P, Gan Y-Z, Shui P, Yu F, Zhang Y, Chen S-L, et al. 3D shape segmentation via shape fully convolutional networks. Comput Graph 2018;76:182–92.
[55] Xu H, Dong M, Zhong Z. Directionally convolutional networks for 3D shape segmentation. In: 2017 IEEE international conference on computer vision. 2017, p. 2717–26.
[56] Guo K, Zou D, Chen X. 3D mesh labeling via deep convolutional neural networks. ACM Trans Graph 2015;35:1–12.
[57] Qiao Y-L, Gao L, Yang J, Rosin PL, Lai Y-K, Chen X. Learning on 3D meshes with Laplacian encoding and pooling. IEEE Trans Vis Comput Graphics 2022;28:1317–27.
[58] Sidi O, van Kaick O, Kleiman Y, Zhang H, Cohen-Or D. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. ACM Trans Graph 2011;30(6):1—10.