Neurocomputing 514 (2022) 328–340

Contents lists available at ScienceDirect

Neurocomputing

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

3D mesh transformer: A hierarchical neural network with local shape tokens ☆

Yu Chen [a,*], Jieyu Zhao [a,*], Lingfeng Huang [a], Hao Chen [a]

[a] Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo 315000, China

ARTICLE INFO

Article history:
Received 12 October 2021
Revised 14 May 2022
Accepted 24 September 2022
Available online 29 September 2022
Communicated by Zidong Wang

Keywords:
self-attention networks
3D mesh Transformer
polynomial fitting
surface subdivision
multilayer Transformer

ABSTRACT

Self-attention networks have revolutionized Natural Language Processing (NLP) and are making impressive strides in image analysis tasks such as image classification and object detection. Inspired by this success, we specifically design a novel self-attention mechanism between local shapes and build a shape Transformer. We split the 3D mesh model into shape patches, which we call shape tokens, and provide polynomial fitting representations of these patches as input to the shape Transformer. The shape token encodes local geometric information and resembles the token (word) status in NLP. The simplification of the mesh model provides a hierarchical multiresolution structure, which allows us to realize the feature learning of a multilayer Transformer. We set high-level features formed by the shape Transformer as visual tokens and propose a vector-type self-attention mechanism to construct a 3D visual Transformer. Finally, we realized a hierarchical network structure based on local shape tokens and high-level visual tokens. Experiments show that our fusion network of 3D shape Transformer with explicit local shape context augmentation and 3D visual Transformer with multi-level structural feature learning achieves excellent performance on shape classification and part segmentation tasks.

1. Introduction

Polygon meshes are an efficient representation of 3D geometry and are widely used in computer graphics to represent virtual objects and scenes. However, meshes are challenging for deep learning architectures because of their unordered elements and irregular structures. Unlike images arranged on a regular pixel grid, a 3D mesh is an irregularly distributed collection embedded in a continuous space. Therefore, the 3D mesh is structurally different from the image, which prevents the direct application of deep network designs, such as networks based on discrete convolution operators that have become a computer vision standard, to 3D meshes.

In response to this challenge, various methods for deep learning on 3D point clouds have emerged. Qi et al. [8] pioneered PointNet for feature learning on point clouds by using multilayer perceptrons (MLPs) and max pooling to overcome the difficulty in handling the disordered input of 3D data. Inspired by the remarkable progress made by convolutional neural networks (CNNs) in the field of image processing, many works [46,33,2,51] have considered defining convolution operators for point clouds that can aggregate the features of local points. In these methods, either point clouds were voxelized, the sequence of input points were reordered, or a regular weight matrix was designed to obtain a canonical domain for convolutions. It is undeniable that these works indirectly realized feature learning on irregular three-dimensional structure data. However, processing the mesh as a point cloud ignores the local structure of the model surface. Since the neighborhood and connectivity principles of point clouds are not clearly defined, the distance between vertices can cause semantic ambiguity. There are many other graphics or grid convolution operators that directly address the mesh model. The representative work is MeshCNN [24], which was proposed by Hanocka et al. MeshCNN combines edge-based convolution operators and scaled layers by leveraging its inherent geodesic connections. Since the number of edges exceeds the number of vertices, this method is not suitable for high-resolution mesh models. Moreover, this type of specially designed convolution operation method is not universal and cannot perform feature learning across regions.

☆ This work was supported by the National Natural Science Foundation of China under grant 62071260, the National Natural Science Foundation of Zhejiang Province under grants LZ22F020001 and LY17F030002, and the International Cooperation Projects of Zhejiang Province under Grant No. 2013C24027.

* Corresponding author.
E-mail addresses: chenyu_cycy@126.com (Y. Chen), zhao_jieyu@nbu.edu.cn (J. Zhao), 2011082075@nbu.edu.cn (L. Huang), nbu_chenhao@126.com (H. Chen).

https://doi.org/10.1016/j.neucom.2022.09.138
0925-2312/

Recently, Transformer [47], the dominant framework in natural language processing, has been applied to image vision tasks and has yielded better performance than popular convolutional neural networks [14,11]. The essential reason for the improved performance when using Transformer over CNNs may be related to the following limitations of the convolution operation [50]: (1) In the convolution operation, not all pixels need to be treated equally. For different target tasks, each pixel should have a different importance. (2) The convolution operation has difficulty capturing long-distance or cross-regional contextual relationships. For example, in image processing, each convolution filter is limited to running on a small local area. (3) Convolution is not an efficient process for understanding sparse, high-level semantic concepts and is not conducive to the learning of high-level features.

The core of the Transformer architecture is the self-attention mechanism, which learns the relationship between sequence elements. In contrast to convolutional neural networks that can only process a single element in a sequence and can only process local contexts in translation, the Transformer architecture is good at processing complete sequences, learning long-distance relationships, and being easily parallelized [20]. At the same time, compared with similar convolutional networks in deep learning, Transformers usually require less prior knowledge, and they are generally pretrained on large (unlabelled) datasets [31]. This pretraining step avoids expensive manual annotations but encodes highly expressive and generalizable representations so that rich relationships between objects in a given dataset can be modelled. In addition, the self-attention operator is essentially a set operator. For the irregularity and permutation invariance of 3D representations, Transformer provides a good mechanism to encode the rich relationship between various data points.

As a result, researchers naturally introduced Transformer into three-dimensional processing. For example, the motivation of the work of [58,19] was to utilize the ability of Transformer to learn aggregate functions for 3D data processing. Specifically, [58] proposed a point Transformer, which uses vector attention to learn the weight of each channel, and [19] strengthened the input embedding to better capture the local environment in the point cloud. [35] was the first to take advantage of the non-local characteristics of Transformer for human shape reconstruction. These rather ground-breaking works were the first to extend Transformer to the three-dimensional field. However, the methods described above did not carefully consider or design 3D self-attention mechanism calculations and therefore cannot capture local features in detail. Moreover, the embedding representation of their input is not as interpretable as the tokens in NLP. It is important to note that a Transformer network suitable for 3D mesh models has not yet been proposed.

Inspired by the above work, we propose a new paradigm that can overcome the above limitations and extend Transformer to 3D mesh model processing. More specifically, our goal is to replace 3D points and polygon mesh sequences with language-like descriptors. A sentence containing multiple words (i.e., tokens) can sufficiently express one meaning. Similarly, a feature containing multiple shape tokens is sufficient to represent the entire 3D model. Unlike point clouds and mesh arrays, this shape token-based representation is compact. Therefore, we propose shape Transformers to (a) re-represent the 3D mesh model with shape tokens and (b) find relationships between shape semantic concepts. To eliminate the redundancy of local shapes and reduce the computational burden, we use unsupervised clustering to form typical local shape tokens to perform shape-based self-attention calculations. In addition, we design a generic module called the 3D visual Transformer for the 3D mesh Transformer. The motivation for this module design is that the shape tokens are processed by the shape Transformer to extract the underlying shape features, and the visual tokens are used to extract the high-level semantic concepts in the three-dimensional model through the 3D visual Transformer. Then, the mesh simplified method is used as the carrier to form multilevel pyramid structure feature learning. It should be mentioned that the 3D visual Transformer module can be used separately from the shape Transformer module. Through the above design, the 3D mesh Transformer becomes a network that is interpretable and can learn multilevel features. Therefore, advanced performance in shape classification and part segmentation can be achieved.

The main contributions of this paper are summarized as follows:

- We propose a novel self-attention calculation method based on local shape representation. It allows a mechanism similar to standard 1D self-attention, taking the local shape of the mesh model surface as a token and designing a matching similarity measure for it. Thus, the well-known 1D Transformer suitable for NLP can be adapted to 3D mesh tasks.
- We build a multilevel network structure based on the Poisson sampling algorithm of the 3D mesh model to form pyramid feature learning. The multilevel structure, which has unified pooling and upsampling, can be used for geometric deep learning and can support dense grid prediction tasks. At the same time, to eliminate the redundancy of local shapes and reduce the computational burden, we use unsupervised clustering to form typical local shape labels to perform shape-based self-attention calculations.
- To learn the semantic structure information between high-level features, we design a 3D visual Transformer module based on vector representation for 3D mesh model processing. This module is invariant to arrangement and cardinality, so it is fundamentally suitable for three-dimensional data processing.
- Experiments show that our 3D mesh Transformer with explicit local shape context enhancement and multilevel structure feature learning achieves the most advanced performance in shape classification and part segmentation tasks.

2. Related Work

2.1. Transformer in NLP

Transformer was first applied to natural language processing (NLP) tasks, where it achieved significant improvements [48,12,6,39]. For example, Vaswani et al. [48] completely abandoned network structures such as RNNs and CNNs and only used the attention mechanism to perform machine translation tasks; good results were achieved. This research has made the attention mechanism a recent research hotspot. Devlin et al. [12] introduced a new language representation model called bidirectional encoder representations with Transformers (BERT), which is a two-way encoder representation of Transformers. Unlike recent language representation models, BERT aims to pretrain deep bidirectional representations by jointly adjusting the left and right contexts in all layers. Therefore, only an additional output layer is required to fine-tune the pretrained BERT representation, i.e., there is no need for basic task-specific architecture modifications. When BERT was published, it obtained state-of-the-art performance on 11 NLP tasks. Brown et al. [6] proved that by increasing the number of parameters, the language model can significantly improve the performance of downstream tasks under few shot settings (only task descriptions and a few examples were given), thus concluding that regular fine-tuning operations can be removed.

These Transformer-based models, with their powerful representation capabilities, have achieved major breakthroughs in the NLP field. However, in NLP, the input is ordered, and the words

have basic semantics, while the data representation of the 3D mesh model is disordered, and a single point or surface generally has no semantic meaning. Therefore, to extend Transformer to the three-dimensional field, we design an embedding method based on the local shape so that each input unit has certain geometric structure information.

2.2. Transformer for 2D Vision

Inspired by the success of the self-attention layer and Transformer architecture in the NLP field, some researchers tried to extend Transformer to the field of image recognition. In vision applications, CNNs were once considered the fundamental component [27,43], but Transformer is currently a viable alternative to CNNs. Dosovitskiy et al. [13] proposed vision Transformer (ViT), the first attempt to apply the pure Transformer directly to the image with as little modification as possible. To achieve this, they divided the image into small blocks and converted these blocks into linear embedding sequences as input to the Transformer. They proved that the dependence on CNN is not necessary. In the image recognition task, applying a pure Transformer model to the sequence of image patches can also perform well. He et al. [26] proposed a novel Transformer-based framework, TransFG, to apply Transformer to fine classification tasks. They integrated all the original attention weights of Transformer into the attention map to guide the network to effectively and accurately select distinctive image blocks and calculate the relationship between them. Liu et al. [37] proposed a new visual Transformer module called Swin Transformer, which can be used as the general backbone of computer vision. This method introduces the CNN's local area and hierarchical feature ideas into Transformer. Of course, Transformer is not only suitable for basic image classification problems. Transformers are also used to solve other computer vision problems, such as object detection, semantic segmentation and video recognition [60,40].

Inspired by the local patch structure used in ViT and the basic semantic information in language words, we propose a local patch representation method based on the topological connection of the 3D mesh model surface and design a novel shape embedding representation that allows these local patches to visually represent shape information.

2.3. Transformer for 3D Vision

Unlike 2D images, 3D data are disordered and unstructured, making it challenging to design neural networks to process them. Qi et al. [8] pioneered PointNet for feature learning on point clouds by using multilayer perceptrons (MLPs) and max pooling to overcome the difficulty in handling the disordered input of 3D data. PointNet extracts a global feature from all point cloud data. Obviously, this is different approach from the current popular CNN method of extracting local features layer by layer. Inspired by CNN, the author also proposed PointNet++ [41], which can extract local features at different scales and obtain deep features through a multilayer network structure. Tchapmi et al. [46] proposed SEGCloud, which uses trilinear interpolation to map the convolutional features of 3D voxels to point clouds and maintains global consistency through a fully connected conditional random field. Hermosilla et al. [28] proposed MCCNN to describe convolution as a Monte Carlo integration problem. This method guarantees that the potential nonuniform sample distribution function is fully considered from the perspective of Monte Carlo, thus cleverly avoiding the problem of directly processing disordered structure data. Li et al. [34] proposed learning the X transformation from the input points to simultaneously weight the input features associated with the points and rearrange them into a potentially implicit canonical order. This method applies the element multiplication and sum operation of the typical convolution operator to the X transformation feature. MeshNet [15] uses the extracted information from the triangular mesh data as features and uses the output of the feature space description module and the structure description module as the input of mesh convolution to build a deep learning network that can directly process the mesh model.

With the breakthrough progress of Transformer in the field of NLP, there have been many recent works on extending Transformer to 3D data. Compared with the original self-attention module in Transformer, Wang et al. [19] proposed an improved self-attention using implicit Laplacian and normalization. Zhao et al. [57] designed a point Transformer layer for point cloud processing and designed a hierarchical network structure. Lin et al. [36] used a graph convolution named Graphormer to improve Transformer. This approach only adds a graph convolution module to the attention module to fuse local information, which forms global and local friendly structures, thereby realizing 3D human pose estimation and mesh reconstruction based on a single image. However, these methods do not have careful consideration for the design of the calculation of the 3D self-attention mechanism; therefore, local features cannot be captured in detail. Moreover, the embedding representation of the input is not as interpretable as the token in NLP. It is important to note that a Transformer network suitable for 3D mesh models has not yet been proposed. However, the 3D mesh Transformer model that we propose can overcome the limitations described above.

2.4. Vectorized Self-Attention

Because of the fixed convolution kernel, CNNs cannot adaptively change the content of different pixels in the image. However, the convolution kernel value for each channel can be different, so it can be adaptive to the channel. Many past studies have used the dot product to calculate scalar attention. Although content adaptation is achieved, the value of each channel is the same, that is, channel adaptation is not possible. As a result, research on vector attention has emerged.

Zhao et al.[55] believed that the use of convolutional networks for image recognition tasks is actually to achieve feature aggregation and feature change. Therefore, decoupling the traditional convolution and understanding feature aggregation as the weighted summation of pixel features in a local area was proposed. In addition, the authors proposed using the attention mechanism to automatically generate this weight, thereby increasing the size of the considered local area without increasing the parameter target, which allowed feature aggregation to adapt to each channel. Therefore, a self-attention mechanism was used instead of convolution as a feature aggregation method. Two forms of self-attention were considered: pairwise self-attention and patchwise self-attention. These two forms of the self-attention mechanism were used as the basic block of the network to propose the SAN network structure.

Both pairwise and patchwise self-attention are represented by vector attention [55], which is mainly used to learn the weights of space and channel dimensions. Compared with traditional convolution, the weight of the convolution kernel is a fixed scalar after the learning is completed, and then this scalar is used to multiply each dimension of a point on the feature map. In the pairwise self-attention method, the weight is learned through network mapping, and the calculation result is a vector. Obviously, this method takes into account the weight of features on different channels. Compared with the above pairwise self-attention, patchwise self-attention does not perform pairing calculations for each feature, but the entire area is used to calculate a weight tensor. Patchwise self-attention is similar to considering all the feature vectors in the

local neighbourhood when deriving the attention vector. The authors showed that vector attention performs better than scalar attention on many image recognition tasks.

Inspired by patchwise self-attention, we consider constructing a weight vector in our method to reference and merge the information of all feature vectors in each channel, thereby helping us achieve multilevel feature learning.

3. Method

3.1. Overall Architecture

In this section, we introduce the details of the 3D mesh Transformer, which is illustrated in Fig. 1. For a given mesh model, we first split the input 3D mesh object into patches through the topological connection information. Each patch is treated as a "shape token", and its feature is set as the vector $\boldsymbol{x}$, which is obtained by the novel 3D shape Transformer method. The specific design details of the 3D shape Transformer method are described in Section 3.2. Then, we regard the feature vector $\boldsymbol{x}$ as a visual token and use the 3D visual Transformer to capture the contextual relationship between high-level features. The dimension of the visual token is related to the number of typical local shape tokens in the entire data set, which are representative shapes formed by the unsupervised cluster learning of shape tokens to reduce the computational burden. These local shape tokens are used in the calculation of the similarity between the query and key of the shape Transformer to reduce the computational complexity and make the feature dimension input to the visual Transformer change only due to the complexity of different dataset models. When we change the size of the patch, only the value of the feature is changed, not the dimension of the feature. In our implementation, we use different patch sizes for comparison, i.e., the number of vertices and triangle meshes contained in each partial surface are not the same. The visual token dimension of each patch is 45 + 3 = 48, i.e., the feature dimension obtained by the shape Transformer is 45. Then, the 3D coordinates of the centre point of the patch are spliced.

Under the precursor of the shape Transformer, we construct a residual 3D mesh Transformer network with the 3D visual Transformer module as the core. The shape Transformer uses shape patches as basic elements to promote the exchange of information between local feature vectors. The visual Transformer integrates a self-attention layer, which is a linear projection that can reduce dimensionality and increase the speed of processing, and residual connections to explore the contextual relationship between the features of high-level visual tokens. This feature learning framework is placed into the simplification method of the mesh model and multilevel feature learning based on the Transformer module is finally formed. To verify the effectiveness of this network, we use the learned features to implement the classification and part segmentation of the 3D mesh model.

3.2. 3D Local Shape Transformer

The easiest way to extend Transformer for 3D mesh data is to treat the entire mesh model as a sentence and each vertex or triangle mesh as a word. By implementing coordinate-based point embedding or surface embedding and using self-attention, a simple mesh Transformer can be realized. However, in NLP, each word embedding in a sentence contains basic semantic information. The independent input coordinates of these simple points ignore the relationship between points and contain limited semantic information. Furthermore, the attention mechanism is very effective in capturing global features, but it may ignore local geometric information, which is essential for 3D mesh learning.

In response to the above problems and inspired by both the local patch structure used by ViT and the basic semantic information of words in NLP, we propose a local shape token representation. This representation can describe the shape of local patches on the surface of the 3D mesh model, thereby capturing local geometric features and obtaining semantic information.

The 3D Local Shape Token representation (3DLST) is proposed as an innovative raw representation. It captures the geometry and structure information of a local region. This section introduces LST and its feature learning module 3D shape Transformer to show how to implicitly encode the geometric and structural information of local regions in the mesh function space.

3.2.1. Local Function

We preprocess a mesh model M with N vertices as a series of local patches. The definition of a local patch is as follows: Let vertex v_i be the centre of the patch and then find the top k vertices adjacent to it through the geodesic distance and connection infor-

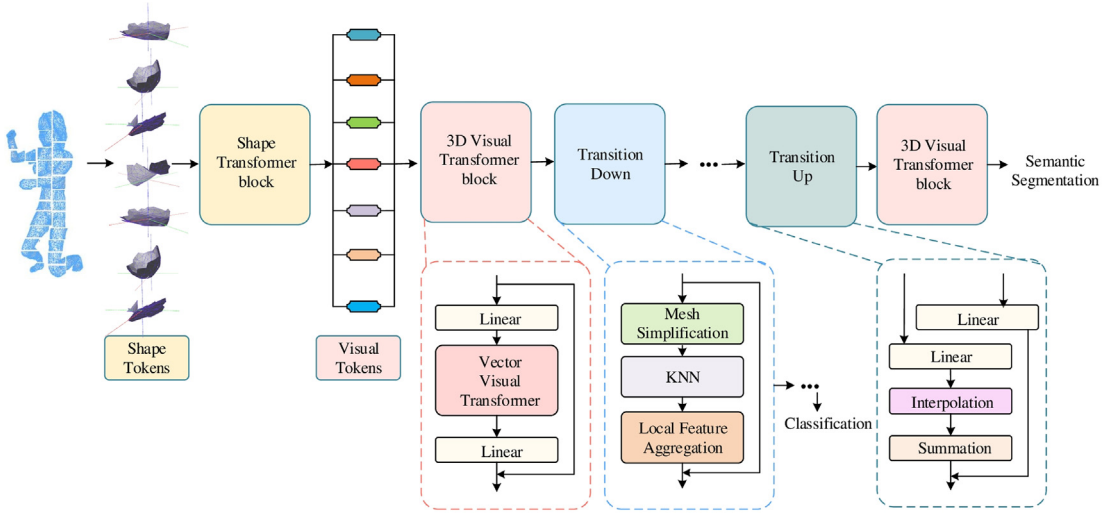

Fig. 1. 3D mesh Transformer. For a given mesh model, we first use a series of local shape surfaces, which we call shape token, to represent the model according to the topological connection information of the model surface. We specifically design a shape Transformer to explore the contextual information between local surfaces. Finally, we regard the output of the shape Transformer as a visual token and use a multilayer vision Transformer for feature learning.

mation. Then, we regard the local mesh consisting of the above k vertices and edges as a patch.

In our method, we employ a higher-order polynomial equation $LF\left(\vec{v}|\vec{\theta}\right)=0$ to describe the shape of each patch, where $LF\left(\vec{v}|\vec{\theta}\right)$ is the continuous function representation of the local surface, $\vec{v}$ is the relative coordinates of the vertices in the patch, and $\vec{\theta}$is the parameter of the polynomial. To better describe the shape of the local patch, we incorporate the approximate geodesic distance d of the vertex relative to the patch centre vertex in the fit, and finally, $\vec{v}=(x,y,z,d)$. The complete local function is as follows:

$$LF\left(\vec{v}|\vec{\theta}\right)=z-\left(\theta_0+\theta_1x+\theta_2y+\theta_3d+\theta_4x^2+\theta_5y^2+\theta_6d^2+\theta_7xy+\theta_8xd+\theta_9yd\right) \tag{1}$$

where the parameter $\vec{\theta}=(\theta_0,\theta_1,\cdots,\theta_9)$ is calculated using the generalized least squares (GLS) method.

Based on $LF\left(\vec{v}|\vec{\theta}\right)$, the local area of v_i is described as a shape. We slide this deformable local mesh window across the surface of the 3D model to obtain the local shape. The union of all the local regions intercepted by the window must cover the entire 3D surface, which means that our model can extract information from the entire 3D shape. Therefore, k cannot be too small. At the same time, an excessively large k will make the shape information extracted by adjacent detection nodes almost the same, reducing the discriminative power of the extracted information. An $LF\left(\vec{v}|\vec{\theta}\right)$ implicitly encodes the geometry and structure information of a local region in the mesh space.

3.2.2. The Acquisition of Typical Shape Tokens

In the last section, the local patch is surfaced to form a shape token, and on this basis, we calculate the self-attention mechanism. However, the self-attention mechanism needs to calculate the similarity between tokens, which requires a large amount of calculation, even more when using three-dimensional data. To reduce the computational burden, we consider replacing one of the query and key sequences with a typical shape token. The typical shape token is essentially a representation of the original local shape; therefore, this is a feasible approach. In this section, we introduce how to obtain a typical shape token.

Our 3DLST method is inspired by the bag-of-features model. The bag-of-features model is modelled on the bag-of-words method in the field of text retrieval, and each image is described as an unordered set of local patch/key point features. Using clustering algorithms (such as $k-means$ clustering) to cluster local features, each cluster centre is seen as a visual word in a vocabulary, which is equivalent to a word in a text search. The visual vocabulary is represented by the code words formed by the corresponding characteristics of the clustering centre (which can be seen as a feature quantization process). All visual words form a visual vocabulary corresponding to a codebook, that is, a collection of code words, and the number of words contained in the dictionary reflects the size of the vocabulary. Each feature in the image is mapped to a word in the visual vocabulary, which can be obtained by calculating the distance between features and then counting how many times each visual word appears. Then, the image can be described as a histogram vector with the same dimension.

In our method, the model is divided into a series of mesh fragments, and a polynomial function is used to fit the shape of the fragments to form a shape patch. After modelling the feature bag, we can describe each mesh object as an unordered collection of shape patches. The clustering method is used to cluster these local shape functions, and each cluster centre can be regarded as a typical local shape, which can form a set of shapes (similar to a visual dictionary) that can be regarded as shape tokens. The detailed process is shown in Fig. 2. As a result, the input of each patch of the model can be described by shape tokens to form a local shape embedding representation in a uniform spatial dimension. Therefore, obtaining the shape tokens is an important part of this study.

We assume that there are c typical local surface shape types in the 3D model data set, corresponding to c shape tokens. Each 3D mesh model can be disassembled into patches with different shapes. We collect shape patches for each type of model in the dataset to form a shape patch set P. The collection of all types of shapes ensures the completeness of the shape set. In this study, a spectral clustering algorithm is used to perform unsupervised learning on the shape set P to obtain c types of typical local surface shapes, that is, c types of shape tokens.

We construct a similarity matrix W for the local shape set P to measure the similarity between local shapes. Given local shape patches P_A and P_B centred on two different vertices v_A and v_B, their surface function representations are $LF_A\left(\vec{v}|\vec{\theta_A}\right)$ and $LF_B\left(\vec{v}|\vec{\theta_B}\right)$, respectively. The distance measurement $dis(P_A,P_B)$ between the local shape patches P_A and P_B is defined by the following equation:

$$dis(P_A,P_B)=\frac{1}{2}\left(\sum_{v_a\in P_A}LF_B^2\left(\vec{v}_a|\vec{\theta_B}\right)+\sum_{v_b\in P_B}LF_A^2\left(\vec{v}_b|\vec{\theta_A}\right)\right) \tag{2}$$

where $LF_B^2\left(\vec{v}_a|\vec{\theta_B}\right)$ represents the distance from shape patch P_A to shape patch P_B, and similarly, $LF_A^2\left(\vec{v}_b|\vec{\theta_A}\right)$ represents the distance from shape patch P_B to shape patch P_A.

We define the two-way shape difference distance between two shape patches in this way to reduce the error caused by shape fitting. We construct a similarity matrix based on the abovementioned difference measure between local shapes and obtain clustering results through spectral clustering.

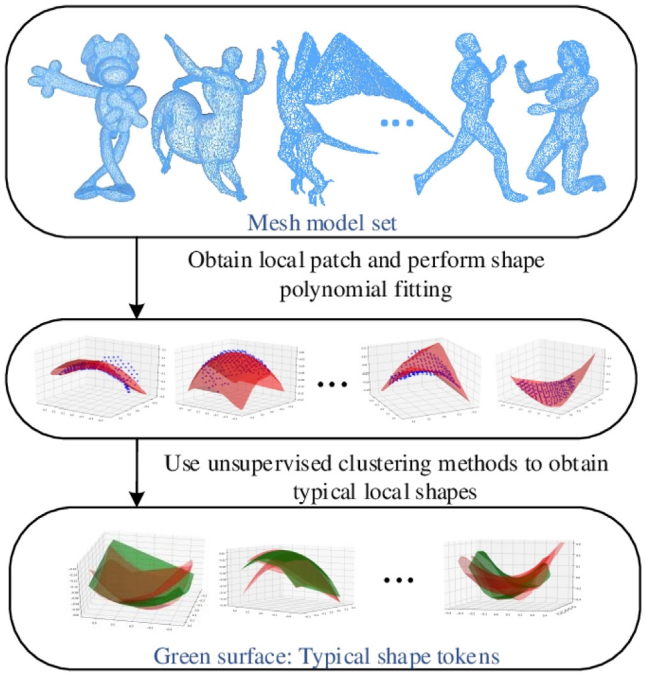

Fig. 2. Typical shape tokens that are obtained through unsupervised learning. In the experiment, one of the models is randomly selected from each type of model, and 10% of the vertices are taken as the centre of the local shape. Then, polynomial fitting is performed on the local shape and unsupervised clustering of the surface set. The final obtained cluster centre is the typical shape token after polynomial fitting.

After obtaining the results of the local mesh shape clustering, the clustering centre, as the "mean" of the same category mesh shape, is difficult to represent by a specific surface.

Therefore, we define the continuous surface representation function of a class of typical local shapes as LF^* and assume that the distribution of this class of typical local shapes in the 3D mesh model dataset obeys the Gaussian distribution, and its distribution variance is σ.

Furthermore, for each cluster S^* in the clustering result, its cluster centre is P^*, and the typical shape function is denoted as LF^*. We take the original local shape P belonging to this class as fitting data and solve for the function parameters of typical shapes, where the parameter σ^2 is calculated as follows:

$$\sigma^2 = \frac{1}{m-1}\sum_{i}^{m} dis(P_i, P^*) \quad P_i \in S^* \tag{3}$$

where m represents the number of original local shapes belonging to shape type S^* in the clustering result.

3.2.3. Shape Attention Mechanism

In NLP, Transformer includes two major structures: an encoder and a decoder, and its core component is the self-attention module. The self-attention mechanism can generate fine attention features based on the global context. The first step in calculating self-attention is to create 3 vectors from the input vector of each encoder. For each token, we create a *Query* vector, a *Key* vector and a *Value* vector. These vectors are generated by multiplying the word embeddings by the 3 training matrices created during the training process. The second step is to score the corresponding words by using the dot product of the query vector and the key vector, and the resulting score set is the attention weight. Then, each value vector is multiplied by the score after softmax to reduce the attention to irrelevant words while keeping the attention degree of the current word unchanged. Finally, the attention feature is defined as the weighted sum of all value vectors with attention weights. Obviously, the output attention feature of each word is related to all input features, so the global context can be learned.

In the model design, we follow the original Transformer [13] as much as possible. One advantage of this intentionally designed simple setup is that it can provide reliable interpretability for the model. The standard vision Transformer reshapes the image $\vec{x} \in \mathbb{R}^{H\times W\times C}$ into a sequence of flattened 2D patches $\vec{x}_p \in \mathbb{R}^{N\times\left(P^2\cdot C\right)}$, where (H, W) is the resolution of the original image and C is the number of channels.

As in [13], Q, K, and V are the query, key and value matrix, respectively. Then, the linear transformation of the input feature F_{in} is generated as follows:

$$Q = F_{in}\cdot W_q, K = F_{in}\cdot W_k, V = F_{in}\cdot W_v \tag{4}$$

where $W_q, W_k and W_v$ are the shared learnable linear transformations. Then, the attention weight can be calculated by the matrix dot product as follows:

$$ATT = Q\cdot K^T \tag{5}$$

These weights are then normalized, and the self-attention output feature F_{out} is defined as the sum of the weighted value vectors by:

$$F_{out} = ATT\cdot V \tag{6}$$

It can be seen from the above equation that in the entire self-attention process, the calculation does not affect the input order, which is very suitable for the disorder and irregular characteristics of 3D data. These conclusions confirm the theoretical possibility of extending Transformer to the three-dimensional field.

In the design of the shape Transformer, we respect the above-mentioned standard self-attention mechanism to the greatest extent. We also design $Q_{mesh}, K_{mesh}, V_{mesh}$ sequences in our method. Essentially, $Q_{mesh}, K_{mesh}, V_{mesh}$ should also represent the shape token as input. However, to reduce the space and time complexity, we define $Q_i \in Q_{mesh}$ as the initial local patch (i.e., the shape token), $K_i \in K_{mesh}$ as the simplified local patch (i.e., the typical shape token), and $V_i \in V_{mesh}$ as the feature value of the local patch. For the 3D mesh model without texture, we set the value of V to 1. In practice, we regard *Key* as the significant eigenfunctions generated after spectral clustering. We perform polynomial fitting on the centre of the class to obtain the significant shapes, which already contain the most representative geometric information. In our experiments, the number of significant eigenfunctions (i.e., typical shapes) is verified as $m = 45$. The specific shape attention mechanism can be found in Fig. 3.

We first designed a similarity measurement mechanism for shape tokens. Unlike VIT, which uses dot products to calculate the similarity between feature vectors, we specifically designed the similarity S_{ATT} between shapes. Based on Eq.(2), given a typical local shape token K_i, its continuous surface representation function is $LF_{K_i}\left(\vec{v}|\vec{\theta}_{K_i}\right)$. Assume that the distribution of this type of local shape in the 3D mesh model dataset obeys a Gaussian distribution, and its distribution variance is σ. Then, in a given three-dimensional mesh model, the probability that the shape of the local shape token q_i is similar to the typical local shape token k_i is:

$$\begin{aligned} S_{att_i} &= Similarity(q_i, k_i) \\ &= P(q_i|\boldsymbol{\theta}_{k_i}, \sigma) \\ &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{\sum_{v_q\in q_i} LF_{k_i}^2\left(\boldsymbol{v}_q|\theta_{k_i}\right)}{2\sigma^2}\right) \end{aligned} \tag{7}$$

To make the attention weight more stable, we take the logarithm of the above equation to represent att_i:

$$S_{att_i} = \ln P(q_i|\theta_{k_i}, \sigma) \tag{8}$$

Finally, we use the regular $F_{out} = S_{ATT}\cdot V$ to obtain the output. In our experiment, for a three-dimensional model containing texture features, V represents its feature value; otherwise, V is set to 1.

3.3. 3D Visual Tokens

In this study, the 3D visual token is mainly obtained through the shape Transformer, and its main purpose is to form a unified representation of high-level features and to explore the contextual relationship between high-level features through the visual Transformer to provide a basis for multilevel feature learning. The high-level feature visual token (VT) of vertex v is defined as:

$$\overrightarrow{VT}(v) = [\beta_1, \beta_0, \ldots, \beta_m] \tag{9}$$

where $\beta_i = S_{att_i}\cdot v_i$. By decomposing the local shape patch P_v under the typical shape tokens, the set of coefficients β_i is able to encode the geometry and structure information of the local region of P_v. Meanwhile, β_i can be regarded as the energy or weight corresponding to the i-th shape tokens.

Then, $\overrightarrow{VT}$ makes the energy distribution of local patches from different meshes comparable, and it is regarded as the raw representation for the 3D mesh Transformer to learn directly. Moreover, for nonrigid three-dimensional models, we assume that the local shape distribution on the model surface is roughly constant as the model pose changes, so $LF\left(\vec{v}|\vec{\theta}\right)$ can withstand the influence of rigid and nonrigid grid transformations, as does $\overrightarrow{VT}$.

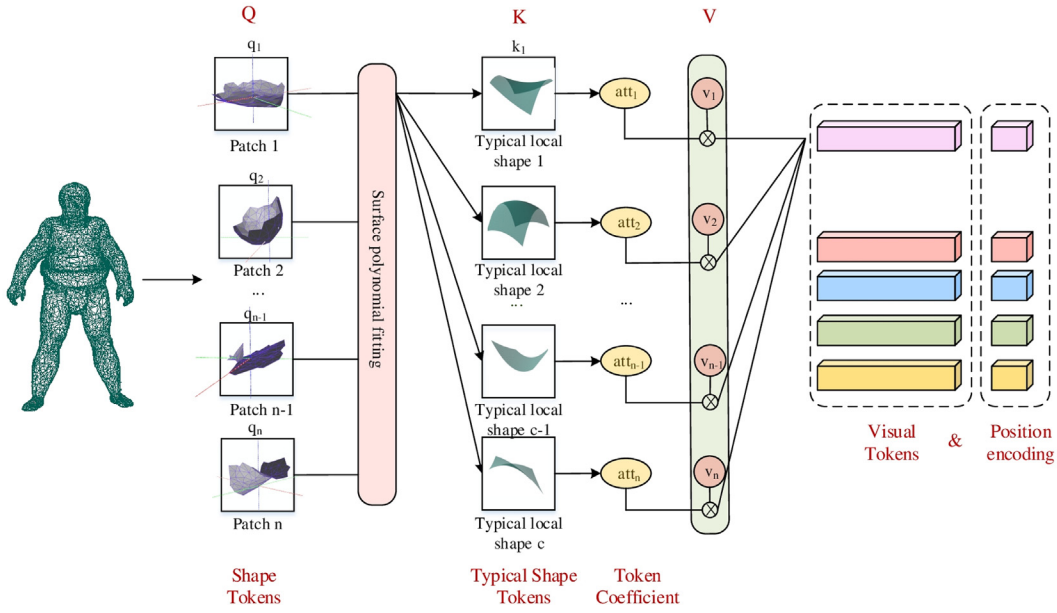

Fig. 3. Shape attention. For a given mesh model, we first split the input 3D mesh object into patches through topological connection information. Each patch is regarded as a "shape token", denoted as q_i. A typical local shape token is a representative shape formed by unsupervised clustering learning of shape tokens to reduce the computational burden. In essence, it still represents the original patch, denoted as k_i. Then, the similarity between the shapes is used to calculate the attention weight. Finally, the weight and the corresponding value vector are multiplied to obtain the output of the shape attention.

The way to obtain the visual token is shown in Fig. 3. For a given mesh model, we first use a series of local shape surfaces, called shape tokens, to represent the model according to the topological connection information of the model surface. We specifically design a shape Transformer to explore the contextual information between local surfaces. Finally, we regard the output of the shape Transformer as a visual token, as shown in Eq.(9).

3.4. Vector Visual Transformer Block

Let $\mathscr{X} = \{\vec{x}_i\}_i$ be a set of local feature vectors in $\overrightarrow{VT}$ with x_i as the centre point. The standard scalar dot product attention layer is expressed as:

$$\vec{y}_i = \sum_{\vec{x}_j \in \mathscr{X}} softmax\Big(q(\vec{x}_i)^\top k(\vec{x}_j) + \delta\Big) v(\vec{x}_j) \tag{10}$$

where $\vec{x}_j$ is the neighbour point of $\vec{x}_i$ in the $\mathscr{X}$ set, $\vec{y}_i$ is the output feature corresponding to the input of $\vec{x}_i, q, k$ and v are the learnable weight transformation matrices in the attention mechanism, which are used to implement point-by-point feature transformation, and δ is used to encode the position. The weight value calculated by $q(\vec{x}_i)^\top k(\vec{x}_j)$ is normalized by the softmax function. It is not difficult to see that the standard scalar attention layer calculates the scalar product between the features transformed by q and k and uses the output as the attention weight to aggregate the features transformed by v.

The biggest difference between vector attention and scalar attention is that in vector attention, the attention weight is a set of vectors. Vector attention is more expressive since it supports adaptive modulation of individual feature channels, not just whole feature vectors.

Visual vector attention $\overrightarrow{V_{ATT}}$ can be expressed as follows:

$$\overrightarrow{V_{attx_j}} = \gamma(q(\vec{x}_i), k(\vec{x}_j)) \tag{11}$$

where the relation function γ is used to discover the similarity between the vectors $q(\vec{x}_i), k(\vec{x}_j)$. For example, subtraction between vectors can be applied as a relational function. Due to the complex spatial relationship of 3D models, we are eager to learn more spatial relationships through vector self-attention. Therefore, in the experiment, for the relationship function between vectors, we used summation, subtraction, concatenation, Hadmard product and dot product to realize the calculation of attention to explore their influence on the experimental results. The corresponding comparison results are shown in Table 5.

Position coding, which allows operators to adapt to the local structure in the data, plays an important role in self-attention [48]. Unlike 2D images, 3D data have their own position coordinates, which can be used directly. Compared with the manual definition in the image, for example, based on the sine and cosine functions [56], three-dimensional space coordinates have natural convenience. To achieve a richer spatial relationship, we introduced trainable parameterized position encoding instead of manual encoding. The parameterized position encoding function $\vec{\delta}$ is defined as follows:

$$\vec{\delta} = \eta(\vec{p}_i - \vec{p}_j) \tag{12}$$

where $\vec{p}_i$ and $\vec{p}_j$ are defined as the 3D coordinates of the centre points v_i and v_j, respectively. The position coding function η is implemented by linear mapping and can be learned through training.

Therefore, our final vector-based self-attention calculation equation is as follows:

$$\vec{y}_i = \sum_{\vec{x}_j \in \mathscr{X}(i)} softmax\Big(mlp\Big(\overrightarrow{V_{attx_j}} + \vec{\delta}\Big)\Big) \odot v\vec{x}_j \tag{13}$$

In this equation, we apply the attention mechanism to the local neighbourhood (especially the k nearest neighbours) of $\vec{v}_i$. The reason for this is that our most primitive input uses local shapes as shape tokens. We hope that this practice can be continued in higher-level feature learning, and the features of the entire model can be learned through a multiscale and multilevel network.

Another reason is that there are more mature methods of image analysis using local self-attention networks that provide us with a theoretical basis.

MLP maps the feature vector to the required dimension to facilitate the representation of the feature dimension. It generates the attention vector for feature aggregation. It is worth noting that the three-dimensional scalar and vector self-attention are still set operators. This set can represent the entire signal [48,13] or a partial patch [29,42,56].

3.5. Transition Down Block

For deep neural networks to benefit from local and global information, pooling operations are usually used to reduce (and then increase) the resolution. A key function of our down transition module is to reduce the cardinality of the point set and the face set to reduce the resolution. The design of our transition module is similar to that of the traditional structured pooling layer. In the classification task, we downsample the vertices of the mesh model. In the segmentation task, we downsample the unit triangle mesh of the mesh model.

Random sampling can be used to reduce the number of samples in a dataset and is widely used in the downsampling of 3D models. However, for this nonuniform sampling, certain features will not be faithfully represented at a rough level. Although other methods (such as farthest point sampling) can handle this problem, they have poor scalability. Therefore, in our spatial pooling layer, we use a uniform sampling method called Poisson disk sampling (PSD) [54], which allows variable point density in space. The classification experiment results show that, compared with the farthest point sampling method, the Poisson disk sampling method can improve the accuracy by 2% on average. The PDS is one of the most classic graphics sampling methods. This sampling strategy generates a random and uniformly distributed set of discrete points within a specified space. The constraint condition of the Poisson disk distribution is that the minimum distance between each point and other points is $2r$, and the parameter r is called the Poisson disk radius. While sampling uniformly, we can automatically control the size of the radius according to the sparsity of the three-dimensional model to obtain a fixed number of sampling points. We denote the point set provided as input to the down transition module as M_1 and the output point set as M_2. To gather the feature vectors in M_1 onto M_2, we use an kNN graph on M_1.

3.6. Transition Up Block

To achieve the intensive prediction task of 3D shape semantic segmentation, we learn from the design of U-Net [7,10]. The U-Net structure is similar to FCN, as it is also divided into a downsampling stage and an upsampling stage. However, there are only convolutional layers and pooling layers in this network. The shallower high-resolution layer is used to solve the problem of pixel positioning, and the deeper layer is used to solve the problem of pixel classification so that image semantic level segmentation can be realized.

In our transition up block, we also represent the input triangular mesh set as M_1 and the output triangular grid mesh as M_2, where $M_1 \ll M_2$. We fuse the features on M_1 to M_2. For the missing feature dimensions, trilinear interpolation is used to map the features to a higher resolution triangular mesh set. Finally, these interpolated features from the previous decoder stage are aggregated with the features from the corresponding encoder stage through skip connections to form a complete high-resolution layer feature.

4. Experiments

The interpretability and flexibility of our 3D local shape token allows the 3D mesh Transformer to be applied to a wide range of 3D shape analysis tasks. We perform a quantitative evaluation of the mesh classification and mesh segmentation tasks of the 3D mesh Transformer and compare it with other state-of-the-art alternatives. For semantic segmentation, the final decoder stage produces a feature vector for each triangle mesh in the input mesh set. We apply an MLP to map this feature to the final logits. For classification, we perform global average pooling on the centre point feature of each shape surface to obtain the global feature vector of the entire point set. The design details of the entire experiment can be seen in Fig. 4.

For the classification task, we conduct experiments on the large Manifold40 dataset [52] and on the deformable datasets SHREC10 [23]and SHREC15 [17]. We compare the classification accuracy with the existing advanced methods. For the segmentation task, verification is performed on the Human Body dataset [38] and a schematic diagram of the segmentation effect is shown. Finally, we utilize different values and analyses for the parameters involved in this study, i.e., the size of the shape token, the number of 3D visual Transformer blocks, and the feature dimension of the output layer.

4.1. Classification

4.1.1. Data Preprocessing and Augmentation

To make the network less sensitive to shape scale, we first scale the input to fit inside a unit cube and apply random anisotropic scaling with a normal distribution, where $\mu = 1$ and $\sigma = 0.1$, which follows [24]. For example, some human body shapes are taller but thinner than others. We also notice that some orientations of shapes in the test dataset do not appear in the training dataset, e.g., in the human body dataset [38]. Therefore, for such datasets, we randomly change the orientation of the input data by rotating around the three axes with Euler angles of $0, \pi/2, \pi, or 3\pi/2$.

4.1.2. Manifold40

ModelNet40 [52], which contains 12,311 shapes in 40 categories, is a widely used 3D geometry learning benchmark. However, most of the 3D shapes in ModelNet40 are not watertight or 2-manifolds, which affects the selection and the fit of local shapes. Therefore, we refer to [30], the shapes in the ModelNet40 dataset were reconstructed into the corresponding Manifold40 dataset, where all shapes are closed manifolds.

The Manifold40 dataset is a more challenging dataset due to reconstruction errors and simplified distortion, and the accuracy of all test methods is reduced, as shown in Table 1. Even so, the 3D mesh Transformer is almost superior to all other mesh-based methods on Manifold40. PCT is an algorithm designed for point clouds, so it can be directly applied to the original data set ModelNet40 without data distortion or other issues.

4.1.3. SHREC15

The SHREC15 dataset includes 1,200 three-dimensional mesh models in 50 categories, each with 24 models, and each model has rigid body transformation and nonrigid body transformation. When training the network, 17 three-dimensional models from each category are randomly selected as training samples, and the remainder of the samples are used as test samples.

We summarize our classification results on SHREC15 in Table 2. We compare several methods designed to use different core operators that use these data in different representations. It can be seen that our method outperforms most methods but is slightly worse

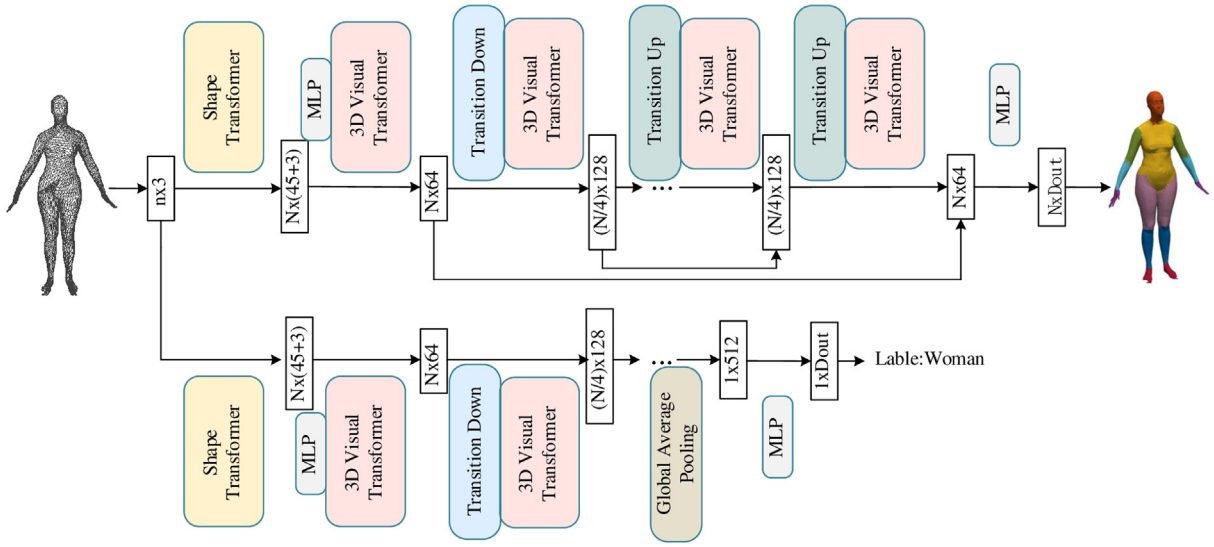

Fig. 4. 3D mesh Transformer networks for shape semantic segmentation (top) and classification (bottom).

Table 1
The classification results obtained by different methods on the ModelNet40 dataset.

Methods	ModelNet40	Manifold40
PointNet++[7]	91.7	87.9
PCT [19]	93.2	92.4
SNGC [21]	91.6	-
MeshNet [15]	91.9	88.4
MeshWalker [32]	92.3	90.5
SubdivNet[30]	-	91.5
Our method	-	**93.6**

Table 2
The classification results obtained by different methods on the SHREC15 dataset.

Methods	Input	Acc
SVM + HKS [5]	feature	56.9
SVM + WKS [3]	feature	87.5
Shape-DNA [44]	point	64.55
cShape-DNA [18]	point	76.21
GPS-embedding [45]	point	75.13
PointNet [8]	point	69.4
PointNet++ [7]	point	60.2
MeshNet[16]	mesh	90.4
MeshCNN[25]	mesh	91.7
MeshCaps [59]	mesh	93.8
SpiderCNN [53]	mesh	95.8
3D MeshConv [9]	mesh	97.3
3D MeshConv (without MSF)[9]	mesh	95.2
Our method	**mesh**	**96.9**

than MeshConv. This is because MeshConv is not a traditional convolution operation. This model supplements the spatial correlation information of the local shape in the deformable target by adopting homogenization-aware Markov stationary features (MSFs), which essentially results in the construction of the spatial distribution relationship between all local shapes. Therefore, MeshConv makes up for the shortcomings of ordinary convolution and achieves similar functions to Transformer. We also list the experimental performance of the pure convolution method in MeshConv, and it can be seen that our method outperforms MeshConv without modelling the global cross-region contextual spatial relationship. It is found that the ability of the Transformer layer to model the contextual relationship between all local shapes is very important in the processing of 3D models.

4.1.4. SHREC10

The SHREC10 dataset includes 200 nonrigid 3D mesh models in 10 categories. There are 20 models in each category, and the same models have rigid body transformation and nonrigid body transformation. In addition, in this dataset, the grid size of each 3D model is relatively uniform. When training the network, 14 three-dimensional models in each category are randomly selected as training samples, and the remaining samples are used as test samples.

We report the average accuracy of the training set and test set in Table 3. The classification accuracy of the 3D mesh Transformer is better than all the methods mentioned. This shows that the proposed method is sufficiently accurate for the SHREC10 nonrigid classification task.

4.1.5. Ablation Study

We now conduct some controlled experiments to check specific decisions in the design of the 3D mesh Transformer. These studies are conducted on the model classification task on the SHREC10 and SHREC15 datasets. The effect of different parameter assignments on the classification accuracy can be seen in Fig. 5. We mainly analysed four parameters, including the number of neighbours of the visual token, the number of 3D visual Transformer blocks, and the feature dimensions of the output layer and the sampling method.

The number of neighbours of the visual token. We first study the setting of the number of neighbours, which is used to determine the local neighbourhood around each visual token. The result is shown in Fig. 5. When the number of neighbours is set to 4, the performance is best. When the neighbourhood is small, the model

Table 3
The classification results obtained by different methods on the SHREC10 dataset.

Methods	Input	Acc
Shape-DNA [44]	point	82.67
cShape-DNA [18]	point	78.50
GPS-embedding [45]	point	87.17
BoW [22]	feature	65.94
BoSCCs [22]	feature	85.99
3D MeshConv [9]	mesh	94.37
3D MeshConv (without MSF)[9]	mesh	91.3
Our method	**mesh**	**97.8**

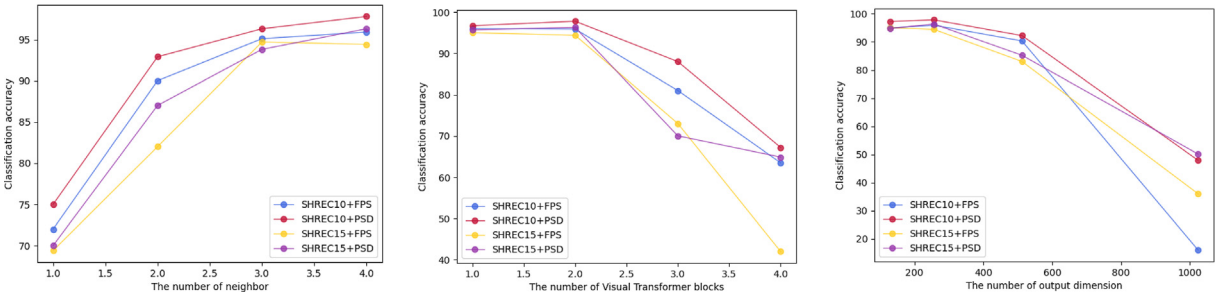

Fig. 5. Parameter analysis of the number of neighbours of the visual token (left), the number of 3D visual Transformer blocks (middle), and the feature dimension of the output layer (right).

may not have enough context to make predictions. When the neighbourhood is large, each self-attention layer will provide a large number of data points, many of which may be farther away and less relevant. This may introduce too much noise in the processing, thereby reducing the accuracy of the model. In the experiment, we set the number of 3D visual Transformer blocks to 2, and the output layer feature dimension is set to 256. On this basis, we assign 1, 2, 3, and 4 to the number of neighbours of the visual token for discussion.

The number of 3D visual Transformer block. We studied the setting of the number of 3D visual Transformer blocks, which is used to determine the depth of the network. The result is shown in Fig. 5. When the number of blocks is set to 2, the performance is the best. When the number of modules is small, the model may not have enough depth for feature learning. When the number of modules is large, there are too many parameters, and the learned features are too redundant, which may reduce the accuracy of the model. This shows that constructing multilevel feature learning is effective.

The number of typical shape token. Typical shape tokens are a series of shape representations in the entire dataset and are proposed as a way to optimize the computational efficiency of Transformers. We analyzed and compared the classification accuracy under different numbers of typical shape tokens, as shown in Fig. 7. We can find that as the number of shape tokens increases, the classification accuracy tends to increase. The classification accuracy is almost optimal for the number of typical shape tokens obtained by unsupervised clustering. We argue that the number of typical shape tokens obtained by unsupervised clustering can represent the local shape of the entire dataset. Too many typical shape tokens require the Transformers to perform more complex self-attention computations, i.e. more 3D models for training. However, due to the limitation of the model set, it will cause a slight decrease in the accuracy.

The feature dimension of the output layer. We conducted ablation studies on the feature dimensions of the output layer. For our experiment, the classification accuracy is higher when the feature dimension is set to 256. This ablation analysis can provide a basis for future experiments.

Sampling method. Usually, the farthest point sampling (FPS) algorithm is used for 3D recognition tasks. However, the Poisson disk sampling (PSD) algorithm that we use is more uniform. As shown in Fig. 5, the algorithm using PSD is on average 1–2% higher in accuracy than the algorithm using FPS.

Position encoding. Position encoding can explicitly model the positional relationship between any two tokens in the Transformer input sequence. Especially in 3D models, we are eager to learn more splicing relationships between shape tokens. Therefore, we make a relevant comparison for the choice of position encoding when the network parameter is 3 neighbours and 1 block. It is worth mentioning that the network parameters are chosen randomly rather than optimally. The results of this experiment are shown in Table 4. When there is no position encoding, the classification accuracy of the model drops significantly. When adding conventional absolute position encoding (i.e., subtraction between the original 3D coordinates), the accuracy is significantly improved. When we add the relative position encoding to the attention weight calculation part, the model classification accuracy reaches its highest value. To further verify the sensitivity of each part to positional encoding in the self-attention mechanism, we simultaneously add the relative position encoding to the attention generation part, the feature part and the above two parts. The results show that the attention generation part has the highest sensitivity to position encoding, and the learned contextual position information is purer, thus achieving the best experimental performance.

Relation function. In regular Transformer, the attention weight is a value obtained by calculating the similarity between input vectors through the dot product. In this study, the relation function $\vec{\delta}$ can produce a vector output so that addition, subtraction, concatenation, Hadamard product and other forms can be introduced to measure the similarity between input vectors. Table 5 reports the comparison of classification results of different relation functions on the SHREC15 dataset. The network parameters are still 3 neighbours and 1 block. It can be seen that the experimental performance using the proposed addition, subtraction, concatenation, and Hadamard product relation functions are all better than the dot product. This finding suggests that vector-type attention weights may be more suitable for processing 3D models than scalar-type attention weights. Vector attention can support the adaptive modulation of a single feature channel, not just the entire feature vector. In particular, the subtractive relation function performs the best experimentally, which may be because in our input vector space, subtraction is more effective in measuring the similarity.

4.1.6. Time and space complexity

Table 6 shows the time and space complexity of our network with several representative methods for classification tasks based on other types of data. Params shows the total number of parameters in the network, and FLOPs shows the number of float opera-

Table 4
Ablation study of the classification results obtained by different position encoding on the SHREC15 dataset.

Position encoding	Acc	Params	FLOPs
none	92.8	0.42 M	0.985G
absolute	94.8	0.39 M	0.788G
relative for attention	**96.3**	**0.42 M**	**0.985G**
relative for feature	93.5	0.42 M	0.985G
relative for both	95.3	0.42 M	0.985G

Table 5
Ablation study of the classification results obtained by different relation functions on the SHREC15 dataset.

Relation	Function	Acc	Params	FLOPs
summation	$q(\vec{x}_i) + k(\vec{x}_j)$	93.1	0.42 M	0.985G
subtraction	$\boldsymbol{q(\vec{x}_i) - k(\vec{x}_j)}$	**95.6**	**0.42 M**	**0.985G**
concatenation	$[q(\vec{x}_i), k(\vec{x}_j)]$	94.0	0.45 M	1.199G
Had. product	$q(\vec{x}_i) \odot k(\vec{x}_j)$	94.4	0.42 M	0.985G
dot product	$q(\vec{x}_i)^\top k(\vec{x}_j)$	92.7	0.53 M	2.389G

tions performed for each input sample, which represent the space and time complexity, respectively.

It can be seen from the comparison that our method has a common problem with the Transformer model, that is, the computational time complexity is high. This is because Transformer needs to calculate the similarity between each input token, so the computational complexity is higher. However, because our method introduces local shape tokens and visual tokens, the parameters of the network are greatly compressed while still ensuring excellent performance in classification experiments. We also provide the time and space complexity of our network as the number of layers increases for reference.

4.2. Segmentation

4.2.1. Human Body Segmentation

We applied our algorithm to the task of semantic segmentation of human models, which were labelled by [38]. As training data, we used 370 models from SCAPE [1], FAUST [4], MIT [49] (excluding two classes that are not suitable for full body segmentation), and Adobe Fuse. All models are manually segmented into eight labels according to the labels in [38]. Our test set is comprised of the 18 models from the SHREC07 dataset in the human category (all sphere-like models). Note that, in contrast with previous works, the training set does not include any models from the SHREC07 dataset, which we used solely for testing. In order to ensure the fairness of the experimental results, we uniformly compare different methods under the condition that the input is 10,000 faces and list the results in Table 7.

We found that the performance of MeshCNN is affected by the resolution. For the competitive method SubdivNet, our segmentation accuracy 92.3 is better than SubdivNet 91.1 when also not optimized using majority voting. On the other hand, we make comparisons for the dimension of the input feature. As can be seen, our method uses 4-dimensional features (3-dimensional coordinates and 1-dimensional approximate geodesic distance), while SubdivNet uses 13-dimensional features (7-dimensional shape description and 6-dimensional pose description). Our method still outperforms SubdivNet with reduced input feature dimensionality.

Some examples of the segmentation results are visualized in Fig. 6. It can be seen that our 3D mesh Transformer method can accurately classify the 8 body parts of the dataset and provide accurate boundaries. This further confirms the effectiveness of our network.

Table 6
Time and space complexity on classification.

Method	Input	Acc	Params	FLOPs
PointNet[8]	point	69.4	3.5 M	0.44G
MeshNet[16]	mesh	90.4	4.251 M	0.509G
MeshCNN[25]	mesh	91.7	1.323 M	0.498G
MeshCaps [59]	mesh	93.8	3.342 M	0.605G
Our method (1 block)	mesh	**96.3**	**0.42 M**	**0.985G**
Our method (2 blocks)	mesh	**96.9**	**0.75 M**	**1.02G**

Table 7
Segmentation results obtained by different methods on the human body dataset.

Methods	Features	Acc
PointNet [8]	3	74.7
PointNet++ [7]	3	82.3
MeshCNN[25]	5	87.7
MeshCNN[25](10000 faces)	5	65.3
Toric Cover[38]	26	88
SubdivNet[30](without majority voting)	13	91.1
SubdivNet[30]	13	93
SubdivNet[30]	7	90.4
Our method	**4**	**92.3**

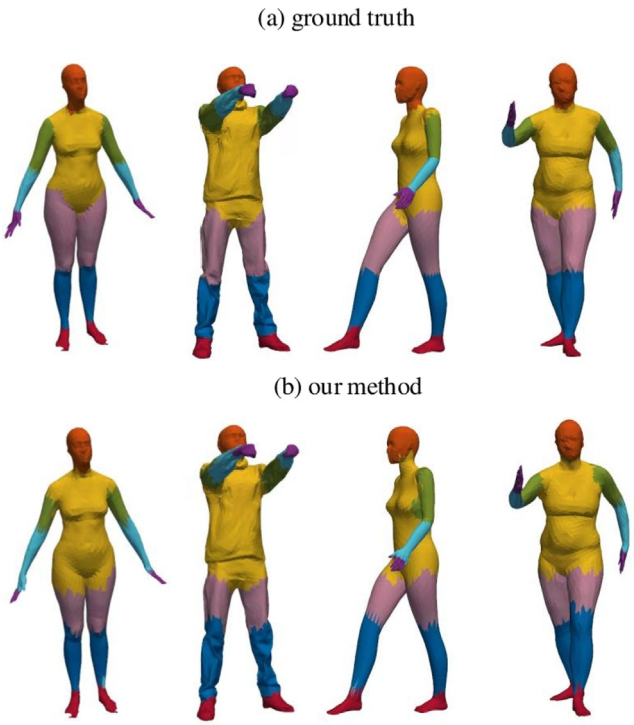

Fig. 6. Segmentation results from the human body dataset. The 3D mesh Transformer can correctly classify the 8 body parts of the dataset and provide accurate boundaries.

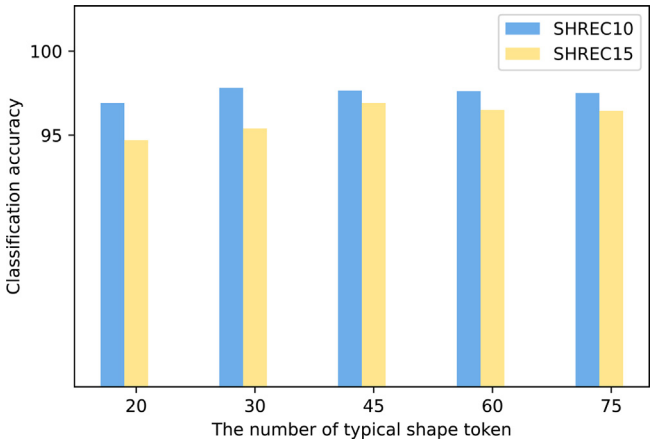

Fig. 7. Classification accuracy for different numbers of typical shape tokens in datasets SHREC10 and SHREC15.

5. Conclusion

In this study, we extend the Transformer model to the three-dimensional domain, which is suitable for learning on unstructured three-dimensional mesh surfaces with irregular domains. We specifically design two token representations, namely, shape tokens and visual tokens. The shape token is similar to the word status in NLP. It encodes the shape information of the local patch, and we regard it as the most primitive input in the network. The visual token summarizes the high-level feature information in the shape Transformer and builds a feature pyramid through the multilayer 3D visual Transformer and transition down module. To verify the effectiveness of our proposed method, we perform classification experiments on a three-dimensional large dataset and a three-dimensional deformable mesh model. We also perform partial shape segmentation experiments on the Human Body dataset. Experiments show that our 3D mesh Transformer with explicit local shape context enhancement and multilevel structure feature learning achieves the most advanced performance in shape classification and part segmentation tasks.

Transformers are suitable for 3D data processing because 3D data are essentially a collection embedded in the metric space, and the self-attention operator at the core of the Transformer network is essentially a collection operator. However, the time complexity and space complexity of this network are large. Reducing the amount of calculation may be the direction of our future research. In addition, the encoder-decoder structure of Transformer support more complex tasks, such as 3D mesh generation. We will extend the 3D mesh Transformer to further applications.

CRediT authorship contribution statement

Yu Chen: Methodology, Formal analysis, Writing - original draft, Visualization. **Jieyu Zhao:** Conceptualization, Writing - review & editing, Funding acquisition. **Lingfeng Huang:** Software, Data curation. **Hao Chen:** Investigation, Resources.

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.

References

[1] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, J. Davis, Scape: Shape completion and animation of people, ACM Trans. Graph. 24 (2005) 408–416, https://doi.org/10.1145/1073204.1073207.

[2] M. Atzmon, H. Maron, Y. Lipman, Point convolutional neural networks by extension operators, ACM Trans. Graph. 37 (2018), https://doi.org/10.1145/3197517.3201301, URL:https://doi.org/10.1145/3197517.3201301.

[3] Aubry, M., Schlickewei, U., Cremers, D.,. The wave kernel signature: A quantum mechanical approach to shape analysis. IEEE International Conference on Computer Vision Workshops, 1626–1633.

[4] Bogo, F., 2014. M.: Faust: Dataset and evaluation for 3d mesh registration, in. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR.

[5] M.M. Bronstein, I. Kokkinos, Scale-invariant heat kernel signatures for non-rigid shape recognition, IEEE Conference on Computer Vision and Pattern Recognition (2010) 1704–1711.

[6] Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., 2020. Language models are few-shot learners. CoRR abs/2005.14165. URL:https://arxiv.org/abs/2005.14165, arXiv:2005.14165.

[7] Charles, R.Q., 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. neural information processing systems, 5099–5108.

[8] Charles, R.Q., Su, H., Kaichun, M., Guibas, L.J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. computer vision and pattern recognition, 77–85.

[9] Y. Chen, J. Zhao, C. Shi, D. Yuan, Mesh convolution: A novel feature extraction method for 3d nonrigid object classification, IEEE Transactions on Multimedia 1–1 (2020), https://doi.org/10.1109/TMM.2020.3020693.

[10] C. Choy, J. Gwak, S. Savarese, 4d spatio-temporal convnets: Minkowski convolutional neural networks, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 3070–3079, https://doi.org/10.1109/CVPR.2019.00319.

[11] Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., Shen, C., 2021. Twins: Revisiting the design of spatial attention in vision transformers, in: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc. pp. 9355–9366. URL:https://proceedings.neurips.cc/paper/2021/file/4e0928de075538c593fbdabb0c5ef2c3-Paper.pdf.

[12] Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota. pp. 4171–4186. URL:https://www.aclweb.org/anthology/N19-1423, doi: 10.18653/v1/N19-1423.

[13] Dosovitskiy, A., Beyer, L., Kolesnikov, A., 2020. An image is worth 16x16 words: Transformers for image recognition at scale. CoRR abs/2010.11929. URL: https://arxiv.org/abs/2010.11929, arXiv:2010.11929.

[14] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale, in: International Conference on Learning Representations. URL:https://openreview.net/forum?id=YicbFdNTTy.

[15] Feng, Y., Feng, Y., You, H., Zhao, X., Gao, Y., 2018. Meshnet: Mesh neural network for 3d shape representation. CoRR abs/1811.11424. URL:http://arxiv.org/abs/1811.11424, arXiv:1811.11424.

[16] Y. Feng, Y. Feng, H. You, X. Zhao, Y. Gao, Meshnet: Mesh neural network for 3d shape representation, in: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI Press, 2019, pp. 8279–8286, https://doi.org/10.1609/aaai.v33i01.33018279, URL:https://doi.org/10.1609/aaai.v33i01.33018279.

[17] Z. Gao, Z. Yu, X. Pang, A compact shape descriptor for triangular surface meshes, Computer-Aided Design 53 (2014) 62–69, https://doi.org/10.1016/j.cad.2014.03.008.

[18] Z. Gao, Z. Yu, X. Pang, A compact shape descriptor for triangular surface meshes, Computer-aided Design 53 (2014) 62–69.

[19] Guo, M., Cai, J., Liu, Z., Mu, T., Martin, R.R., Hu, S., 2020. PCT: point cloud transformer. CoRR abs/2012.09688. URL:https://arxiv.org/abs/2012.09688, arXiv:2012.09688.

[20] M.H. Guo, J.X. Cai, Z.N. Liu, T.J. Mu, Pct: Point cloud transformer, Computational Visual Media 7 (2021) 187–199, https://doi.org/10.1007/s41095-021-0229-5, URL:https://doi.org/10.1007/s41095-021-0229-5.

[21] N. Haim, N. Segol, H. Ben-Hamu, H. Maron, Y. Lipman, Surface networks via general covers, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 632–641, https://doi.org/10.1109/ICCV.2019.00072.

[22] Z. Han, Z. Liu, C.M. Vong, Y. Liu, S. Bu, J. Han, C.L.P. Chen, Boscc: Bag of spatial context correlations for spatially enhanced 3d shape representation, IEEE Transactions on Image Processing 26 (2017) 3707–3720.

[23] Z. Han, Z. Liu, C.M. Vong, Y.S. Liu, S. Bu, J. Han, C.L.P. Chen, Boscc: Bag of spatial context correlations for spatially enhanced 3d shape representation, IEEE Transactions on Image Processing 26 (2017) 3707–3720, https://doi.org/10.1109/TIP.2017.2704426.

[24] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, D. Cohen-Or, Meshcnn: A network with an edge, Acm Transactions on Graphics 38 (2019) 90.1–90.12.

[25] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, D. Cohen-Or, Meshcnn: A network with an edge, ACM Trans. Graph. 38 (2019), https://doi.org/10.1145/3306346.3322959, URL:https://doi.org/10.1145/3306346.3322959.

[26] He, J., Chen, J., Liu, S., Kortylewski, A., 2021. Transfg: A transformer architecture for fine-grained recognition. CoRR abs/2103.07976. URL:https://arxiv.org/abs/2103.07976, arXiv:2103.07976.

[27] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778, https://doi.org/10.1109/CVPR.2016.90.

[28] P. Hermosilla, T. Ritschel, P.P. Vázquez, A. Vinacua, T. Ropinski, Monte carlo convolution for learning on non-uniformly sampled point clouds, ACM Trans. Graph. 37 (2018), https://doi.org/10.1145/3272127.3275110, URL:https://doi.org/10.1145/3272127.3275110.

[29] H. Hu, Z. Zhang, Z. Xie, S. Lin, Local relation networks for image recognition, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 3463–3472, https://doi.org/10.1109/ICCV.2019.00356.

[30] S.M. Hu, Z.N. Liu, M.H. Guo, J.X. Cai, J. Huang, T.J. Mu, R.R. Martin, Subdivision-based mesh convolution networks, ACM Trans. Graph. 41 (2022), https://doi.org/10.1145/3506694, URL:https://doi.org/10.1145/3506694.

[31] S. Khan, M. Naseer, M. Hayat, S.W. Zamir, F.S. Khan, M. Shah, Transformers in vision: A survey, ACM Comput. Surv. (2021), https://doi.org/10.1145/3505244.

[32] A. Lahav, A. Tal, Meshwalker: Deep mesh understanding by random walks, ACM Trans. Graph. 39 (2020), https://doi.org/10.1145/3414685.3417806, URL: https://doi.org/10.1145/3414685.3417806.

[33] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, Pointcnn: Convolution on x transformed points, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA, 2018, pp. 828–838.

[34] Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B., 2018b. Pointcnn: Convolution on x-transformed points, in: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc. URL:https://proceedings.neurips.cc/paper/2018/file/f5f8590cd58a54e94377e6ae2eded4d9-Paper.pdf.

[35] K. Lin, L. Wang, Z. Liu, End-to-end human pose and mesh reconstruction with transformers, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 1954–1963.
[36] Lin, K., Wang, L., Liu, Z., 2021b. Mesh graphormer. CoRR abs/2104.00272. URL: https://arxiv.org/abs/2104.00272, arXiv:2104.00272.
[37] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 9992–10002, https://doi.org/10.1109/ICCV48922.2021.00986.
[38] H. Maron, M. Galun, N. Aigerman, M. Trope, N. Dym, E. Yumer, V.G. Kim, Y. Lipman, Convolutional neural networks on surfaces via seamless toric covers, ACM Trans. Graph. 36 (2017), https://doi.org/10.1145/3072959.3073616, URL: https://doi.org/10.1145/3072959.3073616.
[39] X. Mei, X. Cai, L. Yang, N. Wang, Graph transformer networks based text representation, Neurocomput. 463 (2021) 91–100, https://doi.org/10.1016/j.neucom.2021.08.032, URL:https://doi.org/10.1016/j.neucom.2021.08.032.
[40] D. Neimark, O. Bar, M. Zohar, D. Asselmann, Video transformer network, in: 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2021, pp. 3156–3165, https://doi.org/10.1109/ICCVW54120.2021.00355.
[41] Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc. URL:https://proceedings.neurips.cc/paper/2017/file/d8bf84be3800d12f74d8b05e9b89836f-Paper.pdf.
[42] Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A., Shlens, J., 2019. Stand-alone self-attention in vision models, in: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc. URL:https://proceedings.neurips.cc/paper/2019/file/3416a75f4cea9109507cacd8e2f2aefc-Paper.pdf.
[43] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (2017) 1137–1149, https://doi.org/10.1109/TPAMI.2016.2577031.
[44] M. Reuter, F. Wolter, N. Peinecke, Laplace-beltrami spectra as 'shape-dna' of surfaces and solids, Computer-aided Design 38 (2006) 342–366.
[45] R.M. Rustamov, Laplace-beltrami eigenfunctions for deformation invariant shape representation, in: Proceedings of the Fifth Eurographics Symposium on Geometry Processing, 2007, pp. 225–233.
[46] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, S. Savarese, Segcloud: Semantic segmentation of 3d point clouds, in: International Conference on 3D Vision, 2017, pp. 537–547, https://doi.org/10.1109/3DV.2017.00067.
[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA, 2017, pp. 6000–6010.
[48] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I., 2017b. Attention is all you need, in: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc. URL:https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.
[49] D. Vlasic, I. Baran, W. Matusik, J. Popović, Articulated mesh animation from multi-view silhouettes, in: ACM SIGGRAPH 2008 Papers, Association for Computing Machinery, New York, NY, USA, 2008, https://doi.org/10.1145/1399504.1360696, URL:https://doi.org/10.1145/1399504.1360696.
[50] Wu, B., Xu, C., Dai, X., Wan, A., Zhang, P., Tomizuka, M., Keutzer, K., Vajda, P., 2020. Visual transformers: Token-based image representation and processing for computer vision. CoRR abs/2006.03677. URL:https://arxiv.org/abs/2006.03677, arXiv:2006.03677.
[51] W. Wu, Z. Qi, L. Fuxin, Pointconv: Deep convolutional networks on 3d point clouds, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 9613–9622, https://doi.org/10.1109/CVPR.2019.00985.
[52] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3d shapenets: A deep representation for volumetric shapes, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, Los Alamitos, CA, USA, 2015, pp. 1912–1920, https://doi.org/10.1109/CVPR.2015.7298801, URL:https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298801.
[53] Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y., 2018. Spidercnn: Deep learning on point sets with parameterized convolutional filters. European Conference on Computer Vision.
[54] C. Yuksel, Sample elimination for generating poisson disk sample sets, Computer Graphics Forum 34 (2015) 25–32.
[55] Zhao, H., Jia, J., Koltun, V., 2020a. Exploring self-attention for image recognition, pp. 10073–10082. doi: 10.1109/CVPR42600.2020.01009.
[56] H. Zhao, J. Jia, V. Koltun, Exploring self-attention for image recognition, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10073–10082, https://doi.org/10.1109/CVPR42600.2020.01009.
[57] Zhao, H., Jiang, L., Jia, J., Torr, P., Koltun, V., 2020. Point Transformer. arXiv e-prints, arXiv:2012.09164arXiv:2012.09164.
[58] Zhao, H., Jiang, L., Jia, J., Torr, P.H.S., Koltun, V., 2020. Point transformer. CoRR abs/2012.09164. URL:https://arxiv.org/abs/2012.09164, arXiv:2012.09164.
[59] Y. Zheng, J. Zhao, Y. Chen, C. Tang, S. Yu, 3d mesh model classification with a capsule network, Algorithms 14 (2021) 99, https://doi.org/10.3390/a14030099.
[60] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J., 2021. Deformable detr: Deformable transformers for end-to-end object detection, in: International Conference on Learning Representations. URL:https://openreview.net/forum?id=gZ9hCDWe6ke.

Yu Chen is currently pursuing a Ph.D. degree in pattern recognition and machine intelligence at Ningbo University, Zhejiang, China. Her current research interests include machine learning, pattern recognition, and digital geometry processing.

Jieyu Zhao received his B.S. and M.Sc. degrees from Zhejiang University, China, and his Ph.D. degree from Royal Holloway University of London in 1985, 1988 and 1995, respectively. He is currently a full professor at Ningbo University. His research interests include deep learning, computer vision and environmental surveillance.

Lingfeng Huang is a master's student at Ningbo University, Zhejiang, China. His main research interests include 3D graphics processing and machine learning.

Hao Chen is currently pursuing a Ph.D. degree with the faculty of electrical engineering and computer science at Ningbo University. His main research interests include 3D reconstruction, pattern recognition, and machine learning.