Learning Spectral Dictionary for Local Representation of Mesh

Zhongpai Gao, **Junchi Yan***, **Guangtao Zhai*** and **Xiaokang Yang**
MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
{gaozhongai, yanjunchi, zhaiguangtao, xkyang}@sjtu.edu.cn

Abstract

For meshes, sharing the topology of a template is a common and practical setting in face-, hand-, and body-related applications. Meshes are irregular since each vertex's neighbors are unordered and their orientations are inconsistent with other vertices. Previous methods use isotropic filters or predefined local coordinate systems or learning weighting matrices for each vertex of the template to overcome the irregularity. Learning weighting matrices for each vertex to soft-permute the vertex's neighbors into an implicit canonical order is an effective way to capture the local structure of each vertex. However, learning weighting matrices for each vertex increases the parameter size linearly with the number of vertices and large amounts of parameters are required for high-resolution 3D shapes. In this paper, we learn spectral dictionary (i.e., bases) for the weighting matrices such that the parameter size is independent of the resolution of 3D shapes. The coefficients of the weighting matrix bases for each vertex are learned from the spectral features of the template's vertex and its neighbors in a weight-sharing manner. Comprehensive experiments demonstrate that our model produces state-of-the-art results with a much smaller model size.

1 Introduction

Convolutional operations designed for 3D meshes in deep neural networks have drawn great attention recently. An efficient and effective convolutional operation is desirable and crucial for representation learning of meshes in many 3D tasks, e.g., reconstruction [Tran and Liu, 2018; Gao *et al.*, 2020b], shape correspondence [Groueix *et al.*, 2018], shape synthesis and modeling [Cao *et al.*, 2014b], face recognition [Liu *et al.*, 2018a] and shape segmentation [Đonlić *et al.*, 2017; Kalogerakis *et al.*, 2010], and graphics applications such as virtual avatar [Cao *et al.*, 2014a]. The success of convolutional neural networks (CNN) in the fields where underlying data are Euclidean structured (e.g., images, audio, computed tomography images) has inspired many researchers

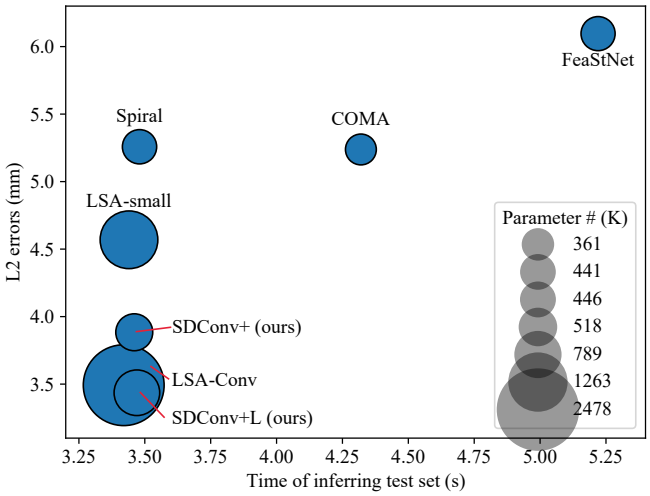

Figure 1: Quantitative evaluation of our SDConv+ against LSA-Conv and LSA-small [Gao *et al.*, 2021], Spiral [Bouritsas *et al.*, 2019], COMA [Ranjan *et al.*, 2018], and FeaStNet [Verma *et al.*, 2018] on the DFAUST dataset for latent size $d = 32$ with the same network architecture, in terms of reconstruction errors, inference time complexity, and parameter size. SDConv+ outperforms LSA-Conv easily by increasing the channel sizes of the network while maintaining a much smaller model size (denoted as SDConv+L).

to adopt CNN on 3D meshes. Different from Euclidean structured data, 3D meshes are non-Euclidean structured and are usually represented as graphs in which the number and orientation of each vertex's neighbors vary from one to another (vertex inconsistency). Thus, an effective convolutional operation that is analogous to CNN meanwhile captures the unique feature of fixed-topology meshes is important for representation learning of meshes.

Many graph convolutional networks have been developed to handle 3D meshes. Defferrard *et al.* [2016] proposed spectral filters that are isotropic using Chebyshev expansion (called *ChebNet*) based on spectral graph theory. Ranjan *et al.* [2018] built convolutional mesh autoencoder (COMA) for fixed-topology 3D meshes upon ChebNet and introduced mesh sampling operations that enable a hierarchical representation to capture non-linear variations of human faces. However, since the convolutional filters are isotropic instead of anisotropic, the expressiveness of ChebNet is limited com-

*Contact Author

pared to its CNN counterpart. Bouritsas *et al.* [2019] introduced anisotropic filters on graph convolutions by formulating a spiral convolution operator (called *SpiralNet*). However, a spiral order of local neighbors cannot resolve the inconsistency between different vertices. Experiments in Table 2 of the recent work [Gao *et al.*, 2021] show that a randomized order of neighbors can even achieve better results.

In this paper, we propose a **s**pectral **d**ictionary based convolutional operation (SDConv) that learns distributed representation for each vertex. We learn a small number of weighting matrices as the bases (i.e., dictionary) and linearly combine these bases to construct weighting matrices for each vertex. The weighting matrices are local structure-aware and can soft-permuted each vertex's neighbors into an implicit canonical order such that conventional CNN with anisotropic filters can be applied in a meaningful way. The coefficients of the bases for each vertex are learned from spectral features of the template's vertex and its neighbors in a weight-sharing manner, so that the parameter size is independent of the template resolution. Furthermore, we introduce a skip connection (i.e., an MLP layer to match the output channel size) in SDConv to further improve its performance, denoted as *SDConv+*.

In line with LSA-Conv [Gao *et al.*, 2021], COMA [Ranjan *et al.*, 2018], SpiralNet [Bouritsas *et al.*, 2019], and SpiralNet++ [Gong *et al.*, 2019], SDConv is also designed for fixed-topology meshes. We evaluate our approach on two fundamental testbeds: autoencoder-based reconstruction and 3D shape correspondence. Note that, meshes with arbitrary topologies can directly be handled by point-cloud based methods [Verma *et al.*, 2018; Li *et al.*, 2018]. We use SDConv+ to build convolutional mesh autoencoder and achieve on par or even better results compared with LSA-Conv with much smaller model size on two 3D shape datasets: human faces (COMA [Ranjan *et al.*, 2018]) and human bodies (DFAUST [Bogo *et al.*, 2017]), as shown in Figure 1.

Our method can be viewed as a combination of global and local methods. In contrast to global methods like COMA [Ranjan *et al.*, 2018] with isotropic filters or predefined local coordinates used in SpiralNet [Bouritsas *et al.*, 2019] using shared model across all the vertices, SDConv personalizes the vertex representation model to enjoy higher local flexibility. Yet compared with the concurrent local structure-aware learning method [Gao *et al.*, 2021] that learns all the weighting matrices for each vertex separately, SDConv notably reduces the model parameter size by a more compact dictionary-based encoding scheme (see Figure 1 for comparison).

The contributions of this paper are summarized below:

1) We propose to learn distributed representation of each mesh vertex, encoded by a **s**pectral **d**ictionary which is embodied as an inherent component in our carefully devised convolutional operation (SDConv) for 3D meshes. SDConv constructs weighting matrices for each vertex from a small number of learnable bases to soft-permute its neighbors based on the local neighboring structure.

2) For more expressive vertex feature learning especially for high-frequency feature, we adopt spectral features with high-frequency information preserved, to learn the corresponding coefficients of weighting matrix bases. Furthermore, an adaptive temperature for softmax is introduced to control the 'softness' of the output distribution of the basis coefficients. Meanwhile, we introduce a skip connection to SDConv, which further improves the performance, denoted as SDConv+.

3) Extensive experiments show that even with much smaller model sizes, SDConv achieves on a par or even better results compared with state-of-the-art methods for 3D shape generations both in terms of time complexity and reconstruction accuracy. The source code has been formally released at: https://github.com/Gaozhongpai/PaiConvMesh.

2 Related Work

Linear 3D morphable models. 3D morphable models (3DMM) are parameterized models of 3D shapes, such as human faces, bodies, hands, etc., and are constructed by performing principal component analysis (PCA) on a training set where 3D meshes share a common template. The widely used 3DMM for faces [Zhu *et al.*, 2015] was built by merging Basel Face Model (BFM) [Paysan *et al.*, 2009] with 200 subjects in neutral expressions and FaceWarehouse [Cao *et al.*, 2014b] with 150 subjects in 20 different expressions. Skinned multi-person linear model (SMPL) [Loper *et al.*, 2015] is the most well-known body model that represents a wide variety of body shapes in natural human poses. MANO [Romero *et al.*, 2017] is a hand model that learned from around 1000 high-resolution 3D scans of human hands in a wide variety of hand poses. Those PCA-based models are commonly used for 3D faces, bodies, and hands reconstruction. The proposed SDConv can be used to build a non-linear 3DMM for 3D shapes with much higher representation power.

Graph neural networks. The popularity of extending deep learning approaches for graph data has been rapidly growing in recent years. ChebNet [Defferrard *et al.*, 2016] and GCN [Kipf and Welling, 2017] generalizes convolution to graphs via Laplacian eigenvectors using fast localized and isotropic filters. GraphSage [Hamilton *et al.*, 2017] samples a fixed number of neighbors and aggregates neighboring features for each vertex. GAT [Veličković *et al.*, 2018] adopts attention mechanisms to learn the relative weights between two connected vertices. MoNet [Monti *et al.*, 2017] introduces vertex neighboring pseudo-coordinates to assigns different weights to the neighbors. FeaStNet [Verma *et al.*, 2018] proposes a graph-convolution operator that learns a weighting matrix dynamically computed from features. LSA-Conv [Gao *et al.*, 2021] proposed to learn weighting matrices for each vertex of the template to soft-permute the vertex's neighbors so that CNNs can be applied. Our SDConv is a concurrent work with LSA-Conv to construct weighting matrices for each vertex. However, instead of learning weighting matrices directly, we adopt the idea of dictionary learning to learn distributed representation for each vertex.

Point neural networks. Point cloud is another type of 3D geometric data. PointCNN [Li *et al.*, 2018] presents a method to learn an $\mathcal{X}$-transformation as a function of input points. KPConv [Thomas *et al.*, 2019] proposes a convolution that takes radius neighbors as input and processes them with weights spatially located by a set of kernel points. Instead of calculating a weighting matrix as a function of inputs in

FeaStNet [Verma *et al.*, 2018], PointCNN [Li *et al.*, 2018], and KPConv [Thomas *et al.*, 2019], our SDConv learns the coefficients of weighting matrix bases for each vertex thanks to the fixed topology of meshes which otherwise is unavailable in cloud data.

3D shape correspondences. 3D shape correspondence is a fundamental task for many computer vision and computer graphics applications, such as shape interpolation, deformation, surface completion, cross-shape texture mapping, etc., [Bogo *et al.*, 2014], [Bogo *et al.*, 2017], where geometric 3D shapes are usually represented by triangle meshes or point clouds. The problem can also be referred as shape registration, alignment, or matching. In line with [Groueix *et al.*, 2018], the 3D shape correspondence task in this paper is to seek reliable correspondences between a 3D shape with randomized vertex order (i.e., point cloud) and a common template shape, which can be achieved by encoding the point cloud shape to a low dimensional global feature using a point cloud encoder and recovering the 3D shape from the global feature to have the same topology with the template using a mesh decoder.

Dictionary learning. Dictionary learning represents data with a linear combination of a small number of bases from a dictionary matrix. The coefficient vector determines how we process the linear combination of these bases with different weights. Recent works have demonstrated that dictionary learning can be built in deep neural networks. Liu *et al.* [2018b] replaced both the fully connected layer and rectified layer with a dictionary learning layer for scene recognition tasks. Mahdizadehaghdam *et al.* [2019] learned a hierarchy of deep dictionaries for image classification tasks. In this paper, we build a spectral dictionary into the convolutional layer for the weighting matrices that can soft-permute each vertex's neighbors into an implicit canonical order.

3 Preliminary

This section briefly recaps LSA-Conv [Gao *et al.*, 2021] for background information and the same mathematical notations are used for consistency. A 3D mesh is represented as $M = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}})$, where $\boldsymbol{\mathcal{V}} = \{1, \ldots, N\}$ is a set of vertices, N is the number of vertices, and $\boldsymbol{\mathcal{E}} \subseteq \boldsymbol{\mathcal{V}} \times \boldsymbol{\mathcal{V}}$ is a set of edges. Vertex has attributes $\boldsymbol{X} \in \mathbb{R}^{D \times N}$, where D is the feature dimension (e.g., $D = 3$ as the 3D space coordinates). For each vertex, $\boldsymbol{\mathcal{N}}_i$ is a set of the one-ring neighborhood of $\boldsymbol{x}_i$ (including itself, also as $\boldsymbol{x}_{i,0}$), where $\boldsymbol{x}_{i,j} \in \boldsymbol{\mathcal{N}}_i$ and $(\boldsymbol{x}_{i,0}, \boldsymbol{x}_{i,j}) \in \boldsymbol{\mathcal{E}}$. We denote the neighbors $\boldsymbol{\mathcal{N}}_i = \{\boldsymbol{x}_{i,0}, \boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,|\boldsymbol{\mathcal{N}}_i|-1}\}$, where $|\boldsymbol{\mathcal{N}}_i|$ is the number of vertex's neighbors and it varies from one vertex to another in a graph.

For each vertex, we construct $\boldsymbol{X}_i = \{\boldsymbol{x}_{i,0}, \boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,K-1}\} \in \mathbb{R}^{D_{in} \times K}$, where the first K neighbors are selected if K is smaller than or equal to $|\boldsymbol{\mathcal{N}}_i|$; otherwise zero-padding is applied, as shown in Fig. 2. The order of each vertex's neighbors $\{\boldsymbol{x}_{i,0}, \boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,K-1}\}$ in $\boldsymbol{X}_i$ is random and unspecified. A learnable weighting matrix is used to soft-permute each vertex's neighbors, denoted as $\boldsymbol{P}_i \in \mathbb{R}^{K \times K}$. The resampled neighbors of each vertex can be obtained by

$$\tilde{\boldsymbol{X}}_i = \boldsymbol{X}_i \boldsymbol{P}_i, \tag{1}$$

where $\tilde{\boldsymbol{X}}_i \in \mathbb{R}^{D_{in} \times K}$ and $\boldsymbol{P}_i$ is a trainable parameter to be adaptive according to the geometric structure of the vertex's neighbors. Then conventional convolution is applied as

$$\boldsymbol{y}_i = \mathrm{vec}(\tilde{\boldsymbol{X}}_i)^\top \boldsymbol{W} + \boldsymbol{b}, \tag{2}$$

where $\boldsymbol{W} \in \mathbb{R}^{(D_{in} \cdot K) \times D_{out}}$ includes D_{out} anisotropic filters, $\boldsymbol{b} \in \mathbb{R}^{D_{out}}$ is the bias, $\boldsymbol{y}_i \in \mathbb{R}^{D_{out}}$ is the output feature vertex corresponding to the input vertex $\boldsymbol{x}_i \in \mathbb{R}^{D_{in}}$, and $\mathrm{vec}(\cdot)$ is a vectorization function which converts a matrix into a column vector. Thus, LSA-Conv is defined as

$$\boldsymbol{y}_i = f\left(\mathrm{vec}(f(\boldsymbol{X}_i \boldsymbol{P}_i))^\top \boldsymbol{W} + \boldsymbol{b}\right), \tag{3}$$

where $f(\cdot)$ is an activation function (e.g., ELU [Clevert *et al.*, 2015]) to introduce non-linearity.

4 Approach

Instead of directly training a local structure-aware weighting matrix for each vertex along with the whole network, we construct the weighting matrices from a small number of learnable bases (i.e., spectral dictionary) and the corresponding coefficients (i.e., distributed representation) to soft-permute each vertex's neighbors.

4.1 Architecture Design for SDConv

LSA-Conv needs to learn weighting matrices for each vertex of the template. For instance, when the number of vertices is $N = 5023$ and the number of neighbors is $K = 9$, the parameter size for the weighting matrices is $N \times K \times K = 406,863$. The model size increases linearly with the number of vertices in the ratio of $K \times K$.

This paper introduces a spectral dictionary method that can significantly reduce the parameter number of LSA-Conv. Learning the weighting matrix for each vertex is unnecessary since the geometric shapes of many vertices are similar to each other. The weighting matrices for all the vertices fall into a small subspace and can be constructed by linear combinations of the low dimensional dictionary (i.e., bases). For meshes with N vertices, we denote $\boldsymbol{P} \in \mathbb{R}^{N \times K \times K}$ for all the weighting matrices that can be factorized as follows,

$$\boldsymbol{P} = \boldsymbol{R}\boldsymbol{D}, \tag{4}$$

where $\boldsymbol{D} \in \mathbb{R}^{B \times K \times K}$ is the B-dimensional dictionary of weighting matrices, $\boldsymbol{R} = \{\boldsymbol{r}_0, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N-1}\} \in \mathbb{R}^{N \times B}$ is the distributed representation (i.e., coefficients) corresponding to the N vertices (subject to $B \ll N$, e.g., $N = 5023, B = 8$).

The coefficients $\boldsymbol{r}_i \in \mathbb{R}^B$ for each vertex are learned from spectral features of the vertex and its neighbors in a weight-sharing manner, as illustrated in Fig. 2. First, we calculate each vertex's feature of the template as

$$\boldsymbol{v}_{i,j} = \{\boldsymbol{t}_{i,0} \oplus (\boldsymbol{t}_{i,0} - \boldsymbol{t}_{i,j}) \oplus \|\boldsymbol{t}_{i,0} - \boldsymbol{t}_{i,j}\|\}, \tag{5}$$

$$\boldsymbol{v}_i = \{\boldsymbol{v}_{i,0} \oplus \boldsymbol{v}_{i,1} \oplus \cdots \oplus \boldsymbol{v}_{i,K-1}\}, \tag{6}$$

where $\boldsymbol{v}_{i,j} \in \mathbb{R}^7$ and $\boldsymbol{v}_i \in \mathbb{R}^{7 \cdot K}$, $\boldsymbol{t}_{i,j} \in \mathbb{R}^3$ is the ith vertex's jth neighbor of the template, $\oplus$ is the concatenation operation, and $\|\cdot\|$ calculates the Euclidean distance between the neighbors and the center point. Then, spectral feature

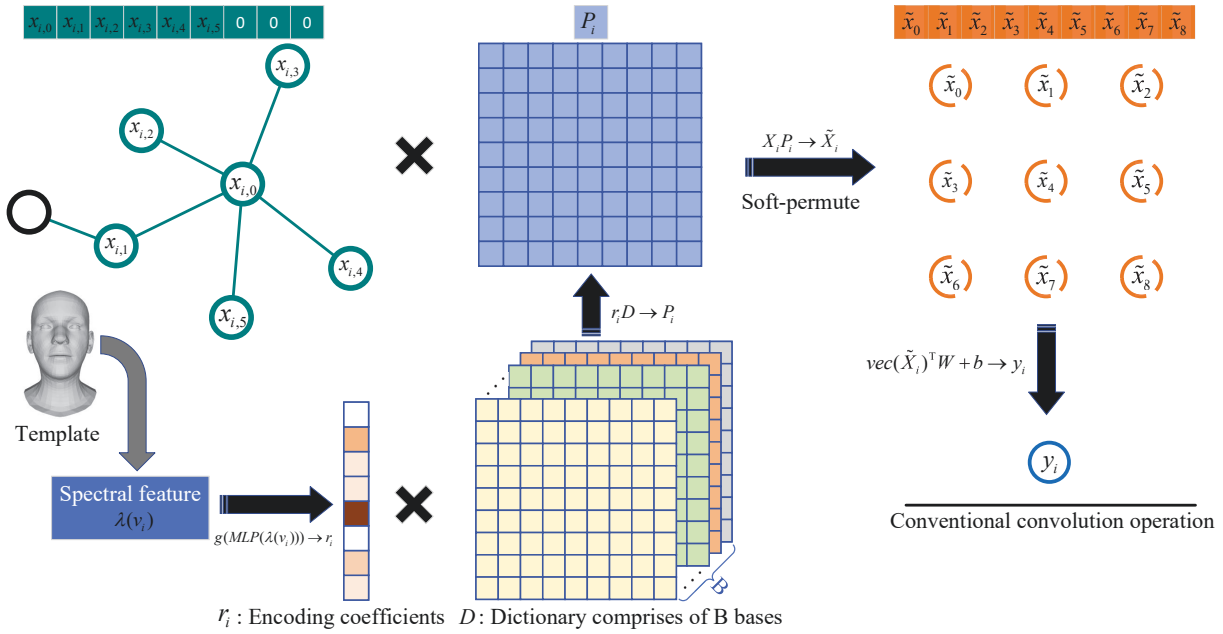

Figure 2: Spectral dictionary convolutional operation (SDConv). $\{\boldsymbol{x}_{i,0}, \boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,5}\} \subseteq \boldsymbol{\mathcal{N}}_i$ are the one-ring neighborhood of $\boldsymbol{x}_i$ (including itself). We construct $\boldsymbol{X}_i = \{\boldsymbol{x}_{i,0}, \boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,K-1}\} \in \mathbb{R}^{D_{in} \times K}$ where K is a predefined neighbor size. $\boldsymbol{v}_i = \{\boldsymbol{v}_{i,0} \oplus \boldsymbol{v}_{i,1} \oplus \ldots \oplus \boldsymbol{v}_{i,K-1}\} \in \mathbb{R}^{K*7}$ is the template vertex feature, followed by a spectral feature mapping function $\lambda(\boldsymbol{v}_i)$. Instead of learning weighting matrices for each vertex, we define B learnable weighting matrices as the bases, $\boldsymbol{D} \in \mathbb{R}^{B \times K \times K}$, and each node's corresponding coefficients, $\boldsymbol{r}_i \in \mathbb{R}^B$, which learns from the spectral features. At last, we construct the weighting matrix for each vertex as $\boldsymbol{P}_i = \boldsymbol{r}_i \boldsymbol{D}$ to soft-permute the vertex's neighbors, which is followed by a conventional convolution operation.

mapping technique is applied to learn high-frequency functions. Spectral feature mapping function λ is used to featurize the vertex feature before passing them through a coordinate-based MLP. The function λ maps the vertex feature $\boldsymbol{v}_i$ to the surface of a higher dimensional hypersphere with a set of sinusoids as:

$$\lambda(\boldsymbol{v}_i) = \{\cos(2\pi \boldsymbol{v}_i \boldsymbol{B}) \oplus \sin(2\pi \boldsymbol{v}_i \boldsymbol{B})\}, \tag{7}$$

where $\boldsymbol{B} \in \mathbb{R}^{7 \cdot K \times M}$ is a fixed random mapping matrix and $2 \cdot M$ is the dimension of the mapped spectral features. At last, the corresponding coefficients are obtained by encoding the spectral features as,

$$\boldsymbol{r}_i = g(MLP(\lambda(\boldsymbol{v}_i))), \tag{8}$$

where $g(\cdot)$ is softmax with adaptive temperature, which is described in Section 4.2. Instead of directly learning the corresponding coefficients $\boldsymbol{r}_i$ as model parameters in [Gao *et al.*, 2021], we only need to learn a small number of sharing weights (i.e., $2 \cdot M \times B$ for a MLP layer) from the spectral features of each vertex and its neighbors across all the template's vertices. Thus, SDConv can be expressed as,

$$\boldsymbol{y}_i = f\left(\text{vec}(f(\boldsymbol{X}_i \cdot \boldsymbol{r}_i \cdot \boldsymbol{D}))^\top \boldsymbol{W} + \boldsymbol{b}\right). \tag{9}$$

SDConv reduces the parameter size significantly. For instance, when $B = 8$, $K = 9$, and $M = 32$, the parameter size for the weighting matrices is $B \times K \times K + 2 \cdot M \times B = 1,160$ compared to $N \times K \times K = 406,863$ for LSA-Conv. Furthermore, a skip connection is added, and the enhanced model is denoted as SDConv+.

$$\boldsymbol{y}_i = f\left(\text{vec}(f(\boldsymbol{X}_i \cdot \boldsymbol{r}_i \cdot \boldsymbol{D}))^\top \boldsymbol{W} + \boldsymbol{b}\right) + MLP(\boldsymbol{x}_i), \tag{10}$$

where $MLP(\boldsymbol{x}_i)$ is the skip connection that matches the input channel to the output channel.

4.2 Adaptive Temperature for Softmax

In Eq. 8, the distributed representation (i.e., coefficients) of the dictionary $\boldsymbol{r}_i$ is produced by using a softmax layer $g(\cdot)$ as,

$$\boldsymbol{r}_{i,j} = g(\boldsymbol{z}_{i,j}) = \frac{\exp(\boldsymbol{z}_{i,j}/T)}{\sum_{k=0}^{B-1} \exp(\boldsymbol{z}_{i,k}/T)}, \tag{11}$$

where $\boldsymbol{z}_i = MLP(\lambda(\boldsymbol{v}_i))$ and T is a hyperparameter named temperature. Using a higher value for T produces a softer probability distribution over the B bases. In this paper, we learns the temperature from the spectral features of the vertex and its neighbors so that the 'softness' of the output distributions is adaptive to the local structure of the vertex. The adaptive temperature is expressed as,

$$T_i = s(MLP(\lambda(\boldsymbol{v}_i))) \cdot (0.1 - 1.0/F) + 1.0/F, \tag{12}$$

where $s(\cdot)$ is Sigmoid function and F is set to 100 such that $T_i \in [0.01, 0.1]$. Since T_i is smaller than 1, the output distributed representation can be more close to binary values than that of the vanilla softmax. This is similar to the meaning of sparse representation.

5 Experiments and Evaluation

We first evaluate the proposed method on two different 3D shape datasets in two tasks: autoencoder-based reconstruction and 3D shape correspondence. For autoencoder-based

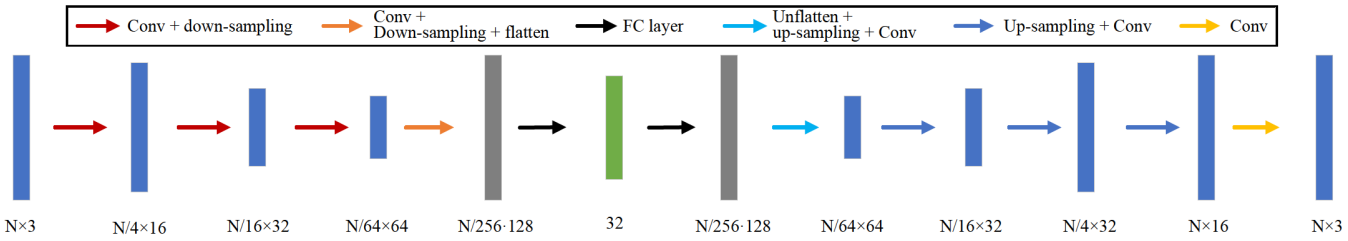

Figure 3: Architecture of the autoencoder-based reconstruction task as the testbed. N is the number of vertices. Conv layer can be COMA, Spiral, LSA-Conv, LSA-small, SDConv, or SDConv+ for comparison. For the 3D correspondence task, the conv layers in the encoder are replaced with PAI-Conv [Gao *et al.*, 2020a] — a point convolutional operation.

reconstruction, the input is 3D meshes with fixed topology. For the task of 3D shape correspondence, the input is point clouds with randomized vertex order. Then, ablation tests are conducted to demonstrate the effectiveness of SDConv.

5.1 Protocol

Datasets. In line with [Gao *et al.*, 2021], we evaluate our model on two datasets: COMA [Ranjan *et al.*, 2018] and DFAUST [Bogo *et al.*, 2017]. COMA is a human facial dataset that consists of 12 classes of extreme expressions from 12 different subjects. The dataset contains 20,466 3D meshes that were registered to a common reference template with 5,023 vertices. DFAUST is a human body dataset that collects over 40,000 real meshes, capturing 129 dynamic performances from 10 subjects. The meshes were also registered to a common reference topology that has 6,890 vertices. Both two datasets are split into training and test set with a ratio of 9:1 and randomly select 100 samples from the training set for validation. All of the 3D meshes are standardized to have a mean of 0 and standard deviation of 1 to speed up the training.

Training. We use Adam [Kingma and Ba, 2014] optimizer with a learning rate of 0.001 and reduce the learning rate with a decay rate of 0.99 in every epoch. The batch size is 32 and the total epoch number is 300. Weight decay regularization is used for the network parameters except for the spectral dictionary of weighting matrices. We initialize the weighting matrix bases with identity matrix according to the experiment in Table 2 of [Gao *et al.*, 2021]. We compare different convolutional operations that are implemented in PyTorch on the same machine with an AMD 3700X @3.6GHz CPU and an NVIDIA RTX2080Ti GPU.

Architecture of autoencoder. We adopt the network architecture from [Gao *et al.*, 2021] and the conv layer can be SDConv, LSA-Conv, Spiral, and COMA for the corresponding methods. The network is an encoder-decoder architecture with a latent vector in the middle, as shown in Figure 3. First, the encoder has four conv layers with downsampling. The conv layers have channel sizes of [3, 16, 32, 64, 128] and meshes are downsampled with ratios of [4, 4, 4, 4]. Then, a fully connected layer outputs the latent vector that represents the 3D mesh. The decoder has five conv layers with upsampling. The conv layers have channel sizes of [128, 64, 32, 32, 16, 3] and meshes are upsampled with ratios of [4, 4, 4, 4, 1]. At last, the decoder outputs the reconstructed 3D mesh.

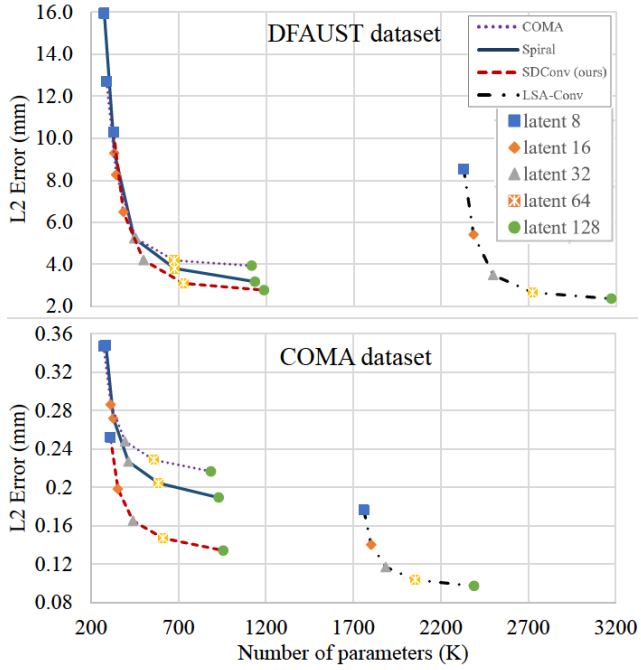

Figure 4: Evaluation of SDConv against peer methods: LSA-Conv, COMA, and Spiral on the test sets of DFAUST and COMA.

Architecture of 3D shape correspondence. We only replace the conv layer of the encoder in Figure 3 with PAI-Conv [Gao *et al.*, 2020a] that is designed for point clouds. The random sampling method is used to downsample point clouds.

5.2 Task I: Autoencoder-based Reconstruction

For the reconstruction task, we compare three existing methods: LSA-Conv [Gao *et al.*, 2021], COMA [Ranjan *et al.*, 2018], and Spiral [Bouritsas *et al.*, 2019] on different dimensionalities of the latent space: 8, 16, 32, 64, and 128 with the same architecture for a fair comparison. As shown in Figure 4, the proposed SDConv achieves the smallest reconstruction errors compared to COMA and Spiral on both COMA and DFAUST datasets for different dimensionalities of the latent space. Compared with LSA-Conv, SDConv has larger reconstruction errors but with much smaller model parameter sizes.

To enhance SDConv, we add a skip connection, denoted as SDConv+ in Eq. 10. As shown in Table 1, given the

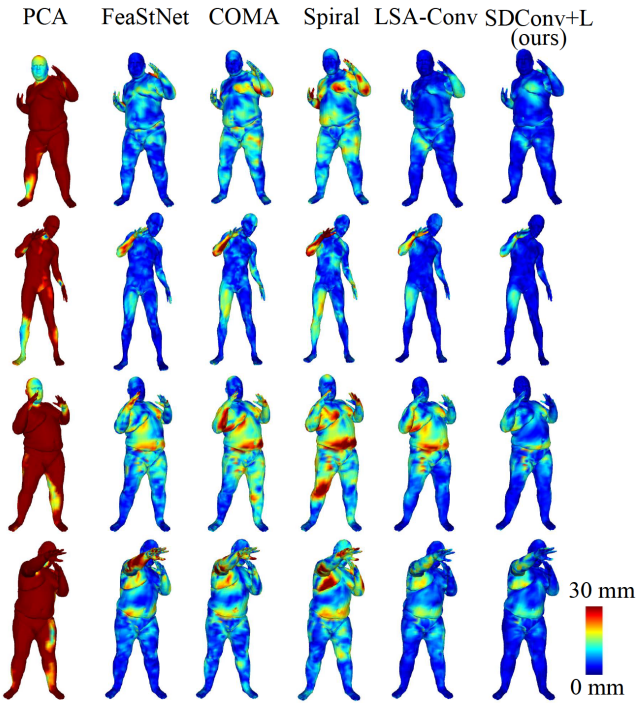

Figure 5: Quantitative evaluation of our SDConv+L (see Figure 1 for the corresponding quantitative results) against LSA-Conv [Gao et al., 2021], Spiral [Bouritsas et al., 2019], COMA [Ranjan et al., 2018], FeaStNet [Verma et al., 2018] and PCA on the DFAUST dataset when the latent size is $d = 32$.

same architecture, SDConv+ outperforms COMA, Spiral, and LSA-small notably. Compared with LSA-Conv, SDConv+ has larger reconstruction errors but with much smaller parameter size. Especially compared with LSA-small — the parameter reduction method proposed by LSA-Conv, SDConv+ achieves better results and has much smaller parameter size. In Figure 5, we qualitatively compare the reconstruction errors of four examples from the test sets of DFAUST datasets when the latent size is $d = 32$. SDConv+ achieves much better visual quality than other methods. The corresponding quantitative results are presented in Figure 1.

For a fair comparison, we adjust the channel sizes of each method to have around the same parameter size. SDConv+ achieves the best results compared with COMA, Spiral, and LSA-small. As presented in Figure 1, in terms of time complexity, SDConv+ is on par with LSA-Conv and Spiral and more efficient than COMA and FeaStNet. In terms of reconstruction accuracy, SDConv+ can even outperform LSA-Conv while maintaining a much smaller model size.

5.3 Task 2: 3D Shape Correspondence

For 3D shape correspondence, we extract the global feature of a 3D shape with randomized vertex order using PAI-Conv [Gao *et al.*, 2020a] which is a convolutional operation designed for point clouds. To compare the expressiveness of different mesh convolutional operations, we recover the 3D shape from the global feature to have the same topology with the template using decoders built by these convolutional operations. We compare our SDConv+ with COMA, Spiral, and LSA-small. As shown in Table 2, SDConv+ achieves the best results, which indicates that SDConv+ has the best representation power to generate 3D shapes. The decoder built by SDConv+ can also be used in 3D shape reconstruction from images and other applications for shape generation.

	DFAUST		**COMA**	
	L2(mm)	parm #	L2(mm)	parm #
PCA	9.977	661K	0.210	482K
	same architecture			
COMA	5.238	361K	0.248	303K
Spiral	5.258	446K	0.227	414K
LSA-Conv	**3.492**	2,478K	**0.117**	1,867K
LSA-small	4.569	1,263K	0.176	986K
SDConv+ (ours)	3.884	518K	0.159	459K
	around same # of parm			
COMA (v2)	5.110	658K	0.198	532K
Spiral (v2)	4.667	647K	0.193	533K
LSA-small (v2)	4.544	644K	0.179	532K
SDConv+ (v2)	**3.673**	643K	**0.148**	531K

	encoder channels	decoder channels
COMA (v2)	[3, 64, 96, 112, 128]	[128, 112, 96, 96, 64, 3]
Spiral (v2)	[3, 32, 64, 64, 128]	[128, 110, 64, 64, 32, 3]
LSA-small (v2)	[3, 16, 32, 64, 128]	[128, 64, 32, 32, 16, 3]
SDConv+ (v2)	[3, 32, 45, 80, 128]	[128, 80, 64, 45, 32, 3]

Table 1: Comparison of reconstruction errors for the models of COMA, Spiral, LSA-Conv, LSA-small, and SDConv+ when latent size $d = 32$. LSA-small is a parameter-reduced version of LSA-Conv. The weighting matrix bases $B = 32$ for LSA-small and SDConv+. For a fair comparison, we increase the channel sizes for COMA (v2), Spiral (v2), SDConv+ (v2) to have around the same parameter size with LSA-small (v2) when $B = 8$, as listed above.

	DFAUST		**COMA**	
	L2(mm)	parm #	L2(mm)	parm #
COMA	20.564	475K	0.341	417K
Spiral	21.398	475K	0.341	417K
LSA-small	17.085	607K	0.312	514K
SDConv+ (ours)	**16.478**	521K	**0.294**	463K

Table 2: Comparison of 3D correspondence errors of COMA, Spiral, LSA-small, and SDConv+ when latent size $d = 32$.

5.4 Ablation Study

The activation function in Eq. 12 is important to calculate the coefficients of the weighting matrix dictionary (i.e., bases). As shown in Table 3, when using vanilla softmax, the reconstruction errors increase largely for both datasets. When using sparsemax, the reconstruction errors are smaller compared to vanilla softmax but are larger than the full model that uses adaptive temperature softmax. Since the temperature of softmax is learned from the spectral features of each vertex and its neighbors, the temperature is adaptive to best optimize the 'softness' of the output distribution for the coefficients of the weighting matrix dictionary.

	DFAUST (mm)	COMA (mm)
Full model	**3.884**	**0.159**
w/ vanilla softmax	4.611	0.175
w/ sparsemax	4.157	0.164
w/o spectral filter	4.387	0.183
w/o skip connection	4.202	0.165

Table 3: Ablation tests of SDConv+ with the latent size $d = 32$ and number of weighting matrix bases $B = 32$.

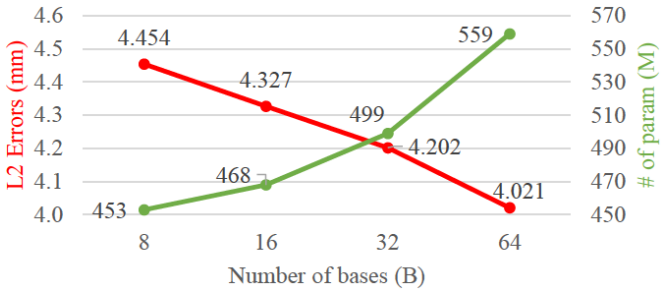

Figure 6: Comparison of numbers of weighting matrix bases for SDConv on DFAUST when latent size $d = 32$. Blue denotes reconstruction error and orange represents model parameter size.

To learn the encoding coefficients, we map the vertex feature to the spectral feature (Eq. 7) that helps preserve high-frequency information. When without the spectral feature mapping function, reconstruction errors increase in both two datasets. Thus, the spectral feature mapping technique can improve regressing the coefficients of the weighting matrix dictionary.

An MLP-based skip connection is added to improve the performance. Without the skip connection, the reconstruction errors increase in both two datasets. The MLP-based skip connection help the model avoid gradient vanishing problem and put more weight on each vertex than its neighbors.

We further investigate the effect of weighting matrix basis size, B, as shown in Figure 6. When we increase the basis size, the reconstruction error decreases and model size increases. In practice, we can choose a basis size for weighting matrices to balance the tradeoff between the reconstruction accuracy and model size.

6 Conclusion

We have proposed a spectral dictionary based convolutional neural network (SDConv) and SDConv+ (with a skip connection) for mesh representation learning and demonstrate their performance on 3D shape generation tasks. We adopt the learnable adaptive-temperature softmax and spectral feature mapping functions to regress the coefficients of the weighting matrix bases (i.e., spectral dictionary) for each vertex. The weighting matrices are constructed by the linear combination of the bases in a weight-sharing manner to soft-permute each vertex's neighbors. Then, a conventional CNN with anisotropic filters is applied.

Compared with methods with either isotropic filters or pre-defined local coordinate systems, our SDConv+ enjoys much higher representation power. Compared with methods that learn weighting matrices directly, SDConv+ achieves competitive or even better results with much smaller model sizes and can be used for applications that involve high-resolution 3D shapes and require high geometric details.

Acknowledgements

This work was supported by National Key Research and Development Program of China (2018AAA0100704), the National Natural Science Foundation of China (U19B2035, 61831105, 61901259, 61972250), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), and China Postdoctoral Science Foundation (BX2019208).

References

[Bogo *et al.*, 2014] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. FAUST: Dataset and evaluation for 3D mesh registration. In *CVPR*, Piscataway, NJ, USA, June 2014. IEEE.

[Bogo *et al.*, 2017] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bodies in motion. In *CVPR*, July 2017.

[Bouritsas *et al.*, 2019] Giorgos Bouritsas, Sergiy Bokhnyak, Stylianos Ploumpis, Michael Bronstein, and Stefanos Zafeiriou. Neural 3D morphable models: Spiral convolutional networks for 3D shape representation learning and generation. In *ICCV*, October 2019.

[Cao *et al.*, 2014a] Chen Cao, Qiming Hou, and Kun Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph.*, 33(4):43:1–43:10, July 2014.

[Cao *et al.*, 2014b] Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. Facewarehouse: A 3D facial expression database for visual computing. *TVCG*, 20(3):413–425, March 2014.

[Clevert *et al.*, 2015] DjorkArné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint*, 2015.

[Defferrard *et al.*, 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *NeurIPS*, pages 3844–3852. Curran Associates, Inc., 2016.

[Gao *et al.*, 2020a] Zhongpai Gao, Guangtao Zhai, Junchi Yan, and Xiaokang Yang. Permutation matters: Anisotropic convolutional layer for learning on point clouds. *arXiv preprint*, 2020.

[Gao *et al.*, 2020b] Zhongpai Gao, Juyong Zhang, Yudong Guo, Chao Ma, Guangtao Zhai, and Xiaokang Yang. Semi-supervised 3d face representation learning from unconstrained photo collections. In *CVPR Workshops*, 2020.

[Gao *et al.*, 2021] Zhongpai Gao, Junchi Yan, Guangtao Zhai, Juyong Zhang, Yiyan Yang, and Xiaokang Yang. Learning local neighboring structure for robust 3d shape representation. In *AAAI*, 2021.

[Gong *et al.*, 2019] Shunwang Gong, Lei Chen, Michael Bronstein, and Stefanos Zafeiriou. Spiralnet++: A fast and highly efficient mesh convolution operator. In *ICCV Workshops*, 2019.

[Groueix *et al.*, 2018] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. 3D-CODED: 3D correspondences by deep deformation. In *ECCV*, September 2018.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeurIPS*, pages 1024–1034. Curran Associates, Inc., 2017.

[Kalogerakis *et al.*, 2010] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3D mesh segmentation and labeling. In *ACM SIGGRAPH*, SIGGRAPH '10, New York, NY, USA, 2010. ACM.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[Li *et al.*, 2018] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution on $\mathcal{X}$-transformed points. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *NeurIPS*, pages 820–830. Curran Associates, Inc., 2018.

[Liu *et al.*, 2018a] Feng Liu, Ronghang Zhu, Dan Zeng, Qijun Zhao, and Xiaoming Liu. Disentangling features in 3D face shapes for joint face reconstruction and recognition. In *CVPR*, June 2018.

[Liu *et al.*, 2018b] Yang Liu, Qingchao Chen, Wei Chen, and Ian Wassell. Dictionary learning inspired deep network for scene recognition. *AAAI*, 32(1), Apr. 2018.

[Loper *et al.*, 2015] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015.

[Mahdizadehaghdam *et al.*, 2019] Shahin Mahdizadehaghdam, Ashkan Panahi, Hamid Krim, and Liyi Dai. Deep dictionary learning: A parametric network approach. *IEEE TIP*, 28(10):4790–4802, 2019.

[Monti *et al.*, 2017] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model CNNs. In *CVPR*, July 2017.

[Đonlić *et al.*, 2017] Matea Đonlić, Tomislav Petković, Stanislav Peharec, Fiona Berryman, and Tomislav Pribanić. On the segmentation of scanned 3d human body models. In *INSHS*, 2017.

[Paysan *et al.*, 2009] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3D face model for pose and illumination invariant face recognition. In *AVSS*, pages 296–301, Sep. 2009.

[Ranjan *et al.*, 2018] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D faces using convolutional mesh autoencoders. In *ECCV*, September 2018.

[Romero *et al.*, 2017] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), November 2017.

[Thomas *et al.*, 2019] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, Francois Goulette, and Leonidas J. Guibas. KPConv: Flexible and deformable convolution for point clouds. In *ICCV*, October 2019.

[Tran and Liu, 2018] Luan Tran and Xiaoming Liu. Nonlinear 3D face morphable model. In *CVPR*, June 2018.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *ICLR*, 2018. accepted as poster.

[Verma *et al.*, 2018] Nitika Verma, Edmond Boyer, and Jakob Verbeek. FeaStNet: Feature-steered graph convolutions for 3D shape analysis. In *CVPR*, June 2018.

[Zhu *et al.*, 2015] Xiangyu Zhu, Zhen Lei, Junjie Yan, Dong Yi, and Stan Z. Li. High-fidelity pose and expression normalization for face recognition in the wild. In *CVPR*, pages 787–796, June 2015.