SpiralNet++: A Fast and Highly Efficient Mesh Convolution Operator

Shunwang Gong[1] Lei Chen[2] Michael Bronstein[1,4] Stefanos Zafeiriou[1,3]
[1]Imperial College London, UK [2]New York University, USA [3]FaceSoft.io [4]Twitter
{shunwang.gong16, m.bronstein, s.zafeiriou}@imperial.ac.uk lc3909@nyu.edu

Abstract

Intrinsic graph convolution operators with differentiable kernel functions play a crucial role in analyzing 3D shape meshes. In this paper, we present a fast and efficient intrinsic mesh convolution operator that does not rely on the intricate design of kernel function. We explicitly formulate the order of aggregating neighboring vertices, instead of learning weights between nodes, and then a fully connected layer follows to fuse local geometric structure information with vertex features. We provide extensive evidence showing that models based on this convolution operator are easier to train, and can efficiently learn invariant shape features. Specifically, we evaluate our method on three different types of tasks of dense shape correspondence, 3D facial expression classification, and 3D shape reconstruction, and show that it significantly outperforms state-of-the-art approaches while being significantly faster, without relying on shape descriptors. Our source code is available on GitHub[1].

Figure 1. Examples of texture transfer from a reference shape in neural pose (**left**) using shape correspondences predicted by SpiralNet++ (**middle**) and SpiralNet (**right**) [25]. Note that we use only 3D coordinates as input features for both methods.

1. Introduction

Geometric deep learning [9] has led to a series of breakthroughs in a broad spectrum of problems ranging from biochemistry [15, 17], physics [12] to recommender systems [29]. This method allows computational models that are composed of multiple layers to learn representations of irregular data structures, such as graphs and meshes. The majority of current works focus on the study of generic graphs [20, 33, 37], whereas it is still challenging to extract non-linear low-dimensional features from manifolds.

A path to 'solving' issues related to 3D computer vision then appears to be paved by defining intrinsic convolution operators. Attempts along this path started from formulating local intrinsic patches on meshes [21, 27, 28], and some other efforts [14, 34] exploit the similar idea of learning the filter weights between the nodes in a local graph neighborhood with utilizing pre-defined local pseudo-coordinate systems over the graphs.

[1]https://github.com/sw-gong/spiralnet_plus

Driven by the significance of the design of kernel weight function, a few questions arise: *Is designing better weight function the vital part of learning representations of manifolds? Can we find more efficient convolution operators without introducing elusive kernel functions and pseudo-coordinates?* It is somewhat intricate to answer if considering the problems defined on generic graphs with varied topologies. These problems, however, are possible to be addressed in terms of meshes, where data [1, 3, 5, 11, 31, 36] are generally aligned.

In this paper, we address these problems by introducing a simple operator, called SpiralNet++, which captures local geometric structure from serializing the local neighborhood of vertices. Instead of randomly generating sequences per epoch [25], SpiralNet++ generates spiral sequences *only once* in order to employ the prior knowledge of fixed meshes, which improves robustness. Since our approach explicitly encodes local information, the model is capable of efficiently learning discriminative features on 3D shapes. We further propose a dilated SpiralNet++ which al-

lows to leverage neighborhoods at multiple scales to achieve detailed captures.

SpiralNet++ is fast, efficient, and easy to apply to various tasks in the domain of 3D computer vision. In our experiments, we bring this operator into three types of challenging problems, *i.e.,* dense shape correspondence, 3D facial expression classification, and 3D shape reconstruction. Without relying on pre-processed shape descriptors or pseudo-coordinate systems, our approach outperforms the competitive baselines by a large margin in all the tasks.

2. Related Work

Geometric deep learning. Geometric deep learning [9] began with attempts to generalize convolutional neural networks for data with an underlying structure that is non-Euclidean. It has been widely adopted to the tasks of graphs and 3D geometry, such as node classification [20, 33], community detection [10], molecule prediction [35], mesh deformation prediction [22], protein interaction prediction [15].

Dense Shape Correspondence. We refer to related surveys [2, 32] on shape correspondence. Ovsjanikov *et al*. [30] formulated a function correspondence problem to find a compact representation that could be used for point-to-point maps. Litany *et al*. [26] took dense descriptor fields defined on two shapes as inputs and established a soft map between the two given objects, allowing end-to-end training. Masci *et al*. [27] proposed to apply filters to local patches represented in geodesic polar coordinates. Boscaini *et al*. [7] proposed the ACNN by using an anisotropic patch extraction method, exploiting the maximum curvature directions to orient patches. Monti *et al*. [28] established a unified framework generalizing CNN architectures to non-euclidean domains. Verma *et al*. [34] proposed a graph-convolution operator of dynamic correspondence between filter weights and neighboring nodes with arbitrary connectivity, which is computed from features learned by the network. Lim *et al*. [25] firstly proposed SpiralNet and applied it on this task, which achieved highly competitive results. However, we observe that because spiral sequences are randomly generated at each epoch, the model is hard to converge and normally requires a larger sequence length as well as high dimensional shape descriptors as input. In order to solve these issues, we present SpiralNet++ that overcomes all of these drawbacks.

3D Facial Expression Classification. Facial expression recognition is a long-established computer vision problem with numerous datasets and methods having been proposed to address it. Cheng *et al*. [11] proposed a high-resolution 4D facial expression dataset, 4DFAB, building a statistical learning model for static and dynamic expression recognition. In this paper, we are the first to introduce SpiralNet++ and other geometric deep learning methods into this task.

Shape Reconstruction. Shape reconstruction is a task that recreates the surface or creates another cross-section [4]. Ranjan *et al*. [31] proposed a convolutional mesh autoencoder (CoMA) based on ChebyNet [13] and spatial pooling to generate 3D facial meshes. Bouritsas *et al*. [8] then integrated the idea of spiral convolution [25] into mesh autoencoder based on the architecture of CoMA, called Neural3DMM. In contrast to SpiralNet [25], they manually selected a reference vertex on the template mesh and defined the spiral sequence based on the shortest geodesic distance from the reference vertex. We argue that it is actually unnecessary to calculate specific spirals but only introducing redundant procedures, since under the assumption of meshes having the same topology, the spirals are already fixed and the same across all the meshes once defined. Additionally, to allow fixed-size spirals for explicit k-disk, they do zero-padding for the vertices that have a smaller spiral length than the average length of k-disk. Intuitively, vertices with a shorter spiral sequence than the average would decrease training efficiency of the weights applied on the concatenated feature vectors, since non-negligible zero paddings always have them not updated. In this paper, our approach addresses these deficiencies and shows the state-of-the-art performance on this task.

3. Our Approach

We assume the input domain is represented as a manifold triangle mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where $\mathcal{V}, \mathcal{E}, \mathcal{F}$ correspond to sets of vertices, edges and faces.

3.1. Main Concept

In contrast to previous approaches [14, 28, 34] which aggregate neighboring node features based on trainable weight functions, our method encodes node features under a explicitly defined spiral sequence, and a fully connected layer follows to encode input features combined with ordering information. It is a simple yet efficient approach. In the following sections, we will elaborate on the definition of spiral sequence and the convolution operation in detail.

3.2. Spiral Sequence

We begin with the definition of spiral sequences, which is the core step of our proposed operator. Given a center vertex, the sequence can be quite naturally enumerated by intuitively following a spiral, as illustrated in Figure 2. The degrees of freedom are merely the orientation within each ring (clockwise or counter-clockwise) and the choice of the starting direction. We fix the orientation to counter-clockwise here and choose an *arbitrary* starting direction. The spirals are pre-computed *only once*.

We first define a k-ring and a k-disk around a center ver-

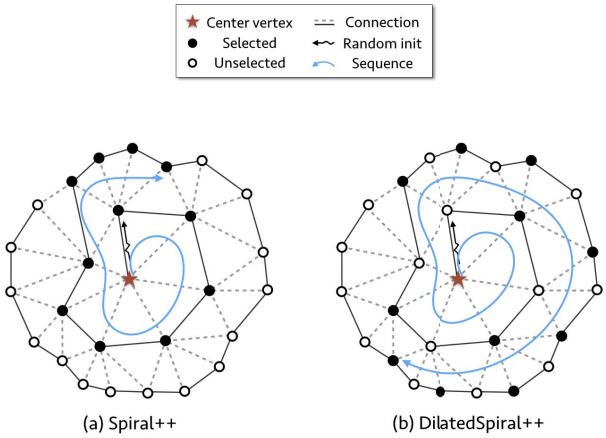

Figure 2. Examples of Spiral++ and DilatedSpiral++ on a triangle mesh. Note that the dilated version supports exponential expansion of the receptive field without increasing the spiral length.

tex v as follows:

$$\begin{aligned} 0{-}\mathrm{ring}(v) &= \{v\}, \\ k{-}\mathrm{disk}(v) &= \cup_{i=0,\dots,k} i{-}\mathrm{ring}(v), \\ (k+1){-}\mathrm{ring}(v) &= \mathcal{N}(k{-}\mathrm{ring}(v)) \backslash k{-}\mathrm{disk}(v), \end{aligned}$$

where $\mathcal{N}(V)$ is the set of all vertices adjacent to any vertex in set V.

Here we denote the spiral length as l. Then $\mathrm{S}(v, l)$ is an *ordered* set consisting of l vertices from a concatenation of k-rings. Note that only part of the last ring will be concatenated to ensure a fixed-length serialization. We define it as follows:

$$\mathrm{S}(v, l) \subset (0{-}\mathrm{ring}(v), 1{-}\mathrm{ring}(v), \dots, k{-}\mathrm{ring}(v)).$$

It shows remarkable advantages to allow the model to learn a high-level feature representation in terms of each vertex in a consistent and robust way when we *freeze* spirals during training. Compared with SpiralNet [25], we credit the major improvement of our approach in terms of speed and efficiency to employing the nature of aligned meshes. Note that since we do not restrict spirals to the scope of a predefined number of rings, we are not involved in performance decays caused by introducing zero-padding [8]. Furthermore, under the assumption of meshes having the same topology, the same vertex across meshes will always have the same spiral sequence regardless of the choice of starting direction, which eases the pain of manually defining the reference point and calculating the start point. By serializing the local neighborhood of vertices we are able to encode relevant information in a straightforward way with very little preprocessing.

3.3. Spiral Convolution

An euclidean CNN [24] designs a two-dimensional kernel sliding on 2D images and maps D input feature maps to E output feature maps.

A common extension of CNNs into irregular domains, such as graphs, is typically expressed as a *neighborhood aggregation* or *message passing* scheme. With $\mathbf{x}_i^{(k-1)} \in \mathbb{R}^F$ denoting node features of node i and $\mathbf{e}_{i,j}^{(k-1)} \in \mathbb{R}^D$ denoting (optional) edge features from node i to node j in layer $(k-1)$, message passing graph neural networks can be described as:

$$\mathbf{x}_i^{(k)} = \gamma^{(k)} \left(\mathbf{x}_i^{(k-1)}, \square_{j \in \mathcal{N}(i)} \, \phi^{(k)}(\mathbf{x}_i^{(k-1)}, \mathbf{x}_j^{(k-1)}, \mathbf{e}_{i,j}^{(k-1)}) \right)$$

where $\mathbf{x}_i^{(k)} \in \mathbb{R}^{F'}$, and $\square$ denotes a differentiable permutation-invariant function, *e.g.*, sum, mean or max, and ϕ denotes a differentiable kernel function. γ represents MLPs. In contrast to CNNs for regular inputs, where there is a clear one-to-one mapping, the main challenge in the case of irregular domains is to define the corerspondence between neighbors and weight matrices which relies on the kernel function ϕ.

Thanks to the nature of the spiral serialization of neighboring nodes, we can define our spiral convolution in an equivalent manner to the euclidean CNNs, easing the pain of calculating the assignment value of $\mathbf{x}_j$ to the weight matrix. We define our spiral convolution operator for a node i as

$$\mathbf{x}_i^{(k)} = \gamma^{(k)} \left(\mathop{\Big\|}_{j \in \mathrm{S}(i,l)} \mathbf{x}_j^{(k-1)} \right)$$

where γ denotes MLPs and $\|$ is the concatenation operation. Note that we concatenate node features in the spiral sequence following the order defined in $\mathrm{S}(i, l)$.

Dilated spiral convolution. With the motivation of exponentially expanding the receptive field without losing resolution or coverage, we define dilated spiral convolution operators. Obviously, spiral convolution operators could immediately gain the power of capturing multi-scale contexts without increasing complexity from uniformly sampling the spiral sequence while keeping the same spiral length, as illustrated in Figure 2.

4. Experiments

In this section, we evaluate our method on three tasks, *i.e.,* dense shape correspondence, 3D facial expression classification, and 3D shape reconstruction. We compare our method against FeaStNet [34], MoNet [28], ChebyNet [13] and SpiralNet [25]. To enable a fair comparison, the model architectures and the kernel size of different convolutions

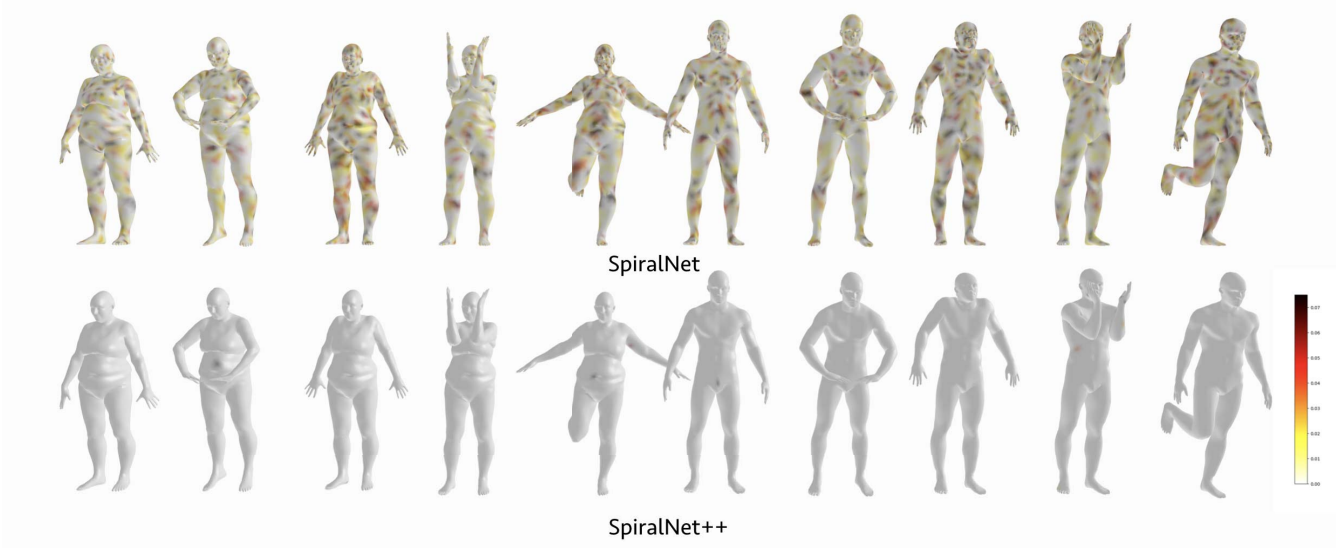

Figure 3. Visualization of pointwise geodesic errors (in % geodesic diameter) of our method and SpiralNet [25] on the test shapes of the FAUST [3] human dataset. The error values are saturated at 7.5% of the geodesic diameter, which corresponds to approximately 15 cm. Hot colors represent large errors.

Method	Acc. (%)	Time/Epoch	# Param
FeaStNet [34]	79.24	3.016s	1.91M
MoNet [28]	86.05	1.962s	1.91M
ChebyNet [13]	98.77	2.634s	1.91M
SpiralNet [25]	72.84	2.756s	1.91M
SpiralNet-LSTM [25]	25.15	3.653s	1.93M
SpiralNet++	**99.88**	**0.98**s	1.91M
SpiralNet-LSTM++	97.86	1.989s	1.93M

Table 1. **Dense shape correspondence** on the FAUST [3] dataset. Test accuracy is the ratio of the correct correspondence prediction with the geodesic error of 0.

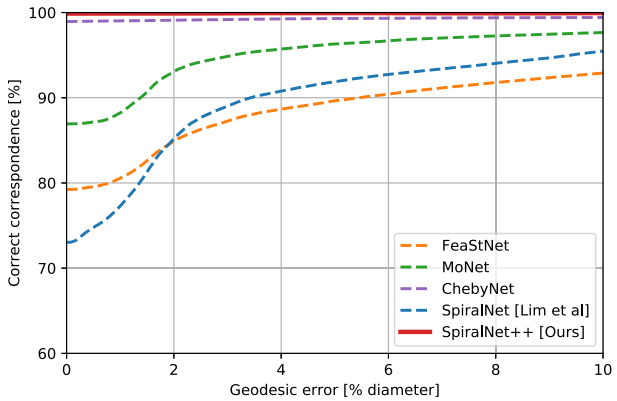

Figure 4. Geodesic error plot of the **shape correspondence** experiments on the FAUST [3] humans dataset. Geodesic error is measured according to the Princeton benchmark protocol [18]. The x axis displays the geodesic error in % of diameter and the y axis shows the percentage of correspondences that lie within a given geodesic error around the correct node.

are the same and fixed, which yields the same level of parameterization. Furthermore, we use raw 3D coordinates as input node features instead of 3D shape descriptors as traditionally used for shape analysis. All the compared methods are with our implementation in order to enforce the same experimental setting except for Neural3DMM [8] that we utilize their code directly. We train and evaluate each method on a single NVIDIA RTX 2080 Ti.

4.1. Dense Shape Correspondence

We validate our method on a collection of three-dimensional meshes solving the task of shape correspondence similar to [6, 27, 28, 34]. Shape correspondence refers to the task of labeling each node of a given shape to the corresponding node of a reference shape [27]. We use the FAUST dataset [3], containing 10 scanned human shapes in 10 different poses, resulting in a total of 100 non-watertight meshes with 6,890 nodes each. The first 80 subjects in FAUST were used for training with the remaining 20 for testing.

Architectuers and parameters. As for all the experiments, we follow the network architecture of [27]. It consists of the following sequence of linear layers (1x1 convolutions) and graph convolutions:

Methods	Anger	Disgust	Fear	Happiness	Sadness	Surprise	Acc. (%)	Time/Epoch	# Param
Baseline [11]	/	/	/	/	/	/	70.27	/	/
FeaStNet [34]	48.40	70.47	63.43	85.86	64.00	92.06	69.89 $\pm$ 1.43	7.364s	69.6k
MoNet [28]	57.87	73.41	63.29	82.86	59.29	89.52	70.29 $\pm$ 3.55	6.457s	69.5k
ChebyNet [13]	65.47	73.18	71.14	**92.00**	67.41	90.48	75.85 $\pm$ 1.47	6.009s	69.4k
SpiralNet++	**68.40**	**82.47**	**71.57**	91.29	**67.65**	**93.97**	**78.59 $\pm$ 0.64**	**3.604**s	69.4k

Table 2. **3D facial expression classification** on the 4DFAB [11] facial expression dataset. We present the test accuracies obtained by all the methods for each expression (*i.e.,* anger, disgust, fear, happiness, sadness and surprise) and all the expressions. *As for the Baseline [11], we use the reported result in their paper.

Lin(16)→Conv(32)→Conv(64)→Conv(128)→Lin(256) →Lin(6890), where the numbers indicate the amount of output channels of each layer. A non-linear activation function, ELU (exponential linear unit), is used after each Conv and the first linear layer. The kernel size or spiral length of all the Convs is 10.

The models are trained with the standard cross-entropy classification loss. We take Adam [19] as the optimizer with the learning rate of 3e-3 (SpiralNet++, MoNet, ChebyNet), 1e-3 (SpiralNet), 1e-2 (FeaStNet), and dropout probability 0.5. As for input features we use the raw 3D XYZ vertice coordinates instead of 544 dimensional SHOT descriptors which was previously used in MoNet [28], SpiralNet [25].

Discussion. In Table 1, we present the accuracy of the exact correspondence (with 0% geodesic error) obtained by SpiralNet++ and other approaches. It shows that our method significantly outperforms all the baselines with 99.88% accuracy and it's counterpart SpiralNet. It should be noted that our method enjoys an extremely fast speed with the training time of 0.98s per epoch in average, which owes to our method exploiting the essence of the fixed mesh topologies. From experiments, We also observed that SpiralNet [25] generally requires around 2500 epochs to converge while it is sufficient for SpiralNet++ to converge within 100 epochs. In Figure 4, we plot the percentage of correspondences that are within a certain geodesic error. In Figure 3, it can be seen that most nodes are classified correctly with our method, which is much better than SpiralNet. Figure 1 visualizes the obtained correspondence using texture transfer.

4.2. 3D Facial Expression Classification

As the second experiment, we address the problem of 3D facial expression classification using the 4DFAB dataset [11], which is a large scale dataset of high-resolution 3D faces. Previous efforts against this task focused on extracting low-dimensional features with PCA and LDA based on manually defined facial landmarks and a multi-class SVM was then employed to classify expressions [11]. Similar to the deep convolutional neural networks used to classify the high-resolution images in the ImageNet [23], we develop an end-to-end hierarchical architecture with our method and other geometric deep learning approaches (*e.g.,* ChebyConv [13], FeaStConv [34], MoNet [28]) to solve this 3D mesh classification problem. Following the experimental setup introduced in [11], we partition the data into 10 folds, and 17 distinct participants in testset are not shown in trainset (with 153 distinct participants). The number of each class is balanced in both training set and test set.

Pooling. The models use a mesh pooling operation based on edge contraction [16]. The pooling operation iteratively contracts vertex pairs to simplify meshes, while maintaining surface error approximation using quadric metrics. The output feature is then directly obtained by the multiplication of input feature with a downsampling transform matrix. We denote a pooling layer using this algorithm with Pool(c), with c being the downsampling factor.

Architectures and parameters. We design the following end-to-end architecture to classify 3D facial expressions: Conv(16) $\rightarrow$ Pool(4) $\rightarrow$ Conv(16) $\rightarrow$ Pool(4) $\rightarrow$ FC(32) $\rightarrow$ FC(6). Dropout with a probability of 0.5 is used before each FC layer. We take a standard cross entropy loss function and ELU activation function. Training is done for 300 epochs with the learning rate of 1e-3, learning rate decay of 0.99 per epoch, L2 regularization of 5e-4, batch size of 32.

It should be noted that raw 3D XYZ coordinates are used as the input, and for MoNet, we use the relative Cartesian coordinates of linked nodes as its pseudo-coordinates. Furthermore, we fixed the same hyperparameters (*i.e.,* kernel size, spiral sequence length or order of the polynomial K) for each convolution, which gives the same size of parameter space of $\mathbb{R}^{K\times \mathrm{C_{in}}\times \mathrm{C_{out}}}$ in terms of each convolution layer.

Discussion All results of the 3D facial expression classification are shown in Table 2. It shows that with our proposed

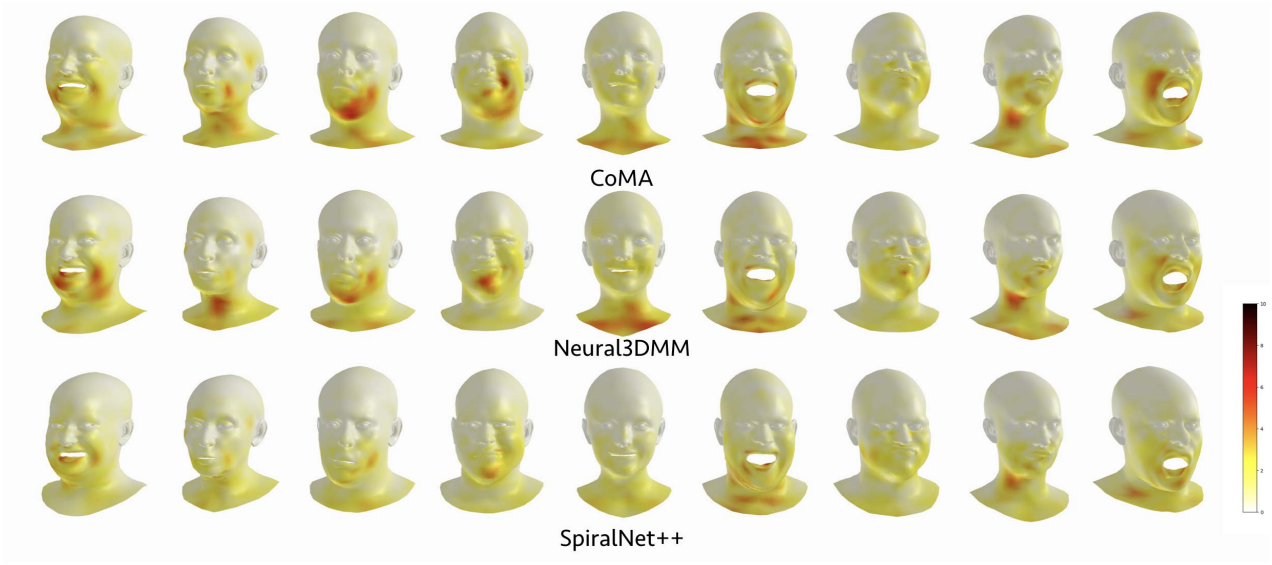

Figure 5. Qualitative results of 3d shape reconstruction in the CoMA [31] dataset. Pointwise error (euclidean distance from the groundtruth) is computed for visualization. The error values are saturated at 10 (millimeters). Hot colors represent large errors.

Method	Mean Error	Median Error	Time/Epoch	# Params
FeaStNet [34]	0.523 ± 0.643	0.297	133.183s	157.9k
MoNet [28]	0.526 ± 0.605	0.353	97.009s	155.4k
CoMA [31]	0.470 ± 0.598	0.263	77.943s	117.5k
ChebyConv (K=9) [13]	0.436 ± 0.562	0.242	86.627s	154.9k
Neural3DMM [8]	0.443 ± 0.560	0.245	107.137	157.0k
SpiralNet++	$\mathbf{0.426 \pm 0.538}$	**0.238**	**30.417**s	154.9k
DilatedSpiralNet++	$\mathbf{0.423 \pm 0.534}$	**0.236**	**29.181**s	154.9k

Table 3. **3D shape reconstruction** experiments results in the CoMA [31] dataset. Errors are in millimeters.

architecture, all of the graph convolution operations outperform the baseline [11]. We credit these improvements to the capacity of learning intrinsic shape features compared to the baseline method. Specifically, our method achieves the highest recognition rate of 78.59% on average. This indicates that SpiralNet++ can be successfully applied to *multi-scale mesh data* improving previous results in this domain. Furthermore, it can be seen that our method is much more faster than all the other approaches.

4.3. 3D Shape Reconstruction

As our largest experiment, we evaluate the effectiveness of SpiralNet++ on an extreme facial expression dataset. We demonstrate that a standard autoencoder architecture with SpiralNet++ allows the synthesis of high-fidelity 3D face with rich expression details. We use the dataset introduced in [31], which consists of 12 classes of extreme expressions, containing over 20,465 3D meshes, each with about 5,023 vertices and 14,995 edges. Following the interpolation experimental setup [8, 31], we divide the dataset into training and test sets with a split ratio of 9:1. We compare our SpiralNet++ against a number of baselines including CoMA [31] and Neural3DMM [8], and furthermore, for the first time, we bring MoNet [28] and FeaStNet [34] into this problem to explore the performance of other intrinsic convolution operations on generative models. It is worth highlighting that in the original work of CoMA [31], they used ChebyNet with $K = 6$. However, in order to have a fair comparison with other experiments, we show both results obtained with $K = 6$ (*i.e.,* CoMA) and $K = 9$. In the end, we evaluate our proposed dilated spiral convolution on this problem.

Pooling and unpooling. The performance of each generative model is closely related to the pooling and unpooling procedures. The same pooling strategy introduced in Section 4.2 is used here. In the unpooling stage, contracted vertices are recovered using the barycentric coordinates of the closet triangle in the decimated mesh [31].

Architectures and parameters. We build a standard autoencoder architecture, consisting of an encoder and a decoder. The encoder includes several convolutional layers interleaved between pooling layers, and one fully connected layer is applied in the end of the encoder to encode non-linear mesh representations. Specifically, the structure is: $3 \times \{\text{Conv}(32) \rightarrow \text{Pool}(4)\} \rightarrow \{\text{Conv}(64) \rightarrow \text{Pool}(4)\} \rightarrow \text{FC}(16)$, with ELU activation function after each Conv layer. The structure of the decoder is the reversed order of the encoder with the replacement of pooling layers to unpooling layers. Note that one more convolutional layer with the output dimensional of 3 should be added to the end of the decoder to reconstruct 3D shape coordinates. Training is done using Adam [19] for 300 epochs with learning rate of 0.001, learning rate decay of 0.99 per epoch and a batch size of 32.

We evaluate all the methods with the same architecture and hyperparameters. The kernel size of each methods is set as 9 in order to keep aligned with Neural3DMM [8], where they chose *1*-hop deriving the spiral length of 9.

Discussion. Table 3 shows mean euclidean errors with standard deviations, median errors and the training time per epoch. Our SpiralNet++ and its dilated version outperform all the other approaches. The result of our proposed dilated spiral convolution validates our assumption, which shows the higher capacity of capturing non-linear low-dimensional representations of 3D shape meshes without increasing parameters. We credit this improvement to its larger receptive field brought by sampling larger input feature space. Moreover, we should stress the remarkable speed of our method. With the same autoencoder architecture, SpiralNet++ is a few times faster than all the other methods. It should be noted that the performance of Neural3DMM is even worse than CoMA when bring weight matrices to the same number, which can be attributed to the fact that model learning is disrupted from introducing non-negligible information (*i.e.*, zero-padding). The performance of Neural3DMM would decrease with the variance of vertex degrees increase. Figure 5 shows the visualization of reconstructed faces in the test set. Larger errors can be seen from the faces generated by CoMA and Neural3DMM, and in particular, it become worse on the faces with extreme expressions. However, SpiralNet++ shows better reconstruction quality in these cases.

5. Conclusions

We explicitly introduce SpiralNet++ to the domain of 3D shape meshes, where data are generally aligned instead of varied topologies, which allows SpiralNet++ to efficiently fuse neighboring node features with local geometric structure information. We further apply this method to the tasks of dense shape correspondence, 3D facial expression classification and 3D shape reconstruction. Extensive experimental results show that our approach are faster and outperform competitive baselines in all the tasks.

6. Acknowledgements

SG and MB were supported in part by the ERC Consolidator Grant No.724228 (LEMAN), Google Faculty Research Awards, Amazon AWS Machine Learning Research grant, and the Royal Society Wolfson Research Merit award. SZ was partially supported by the EPSRC Fellowship DEFORM (EP/S010203/1) and a Google Faculty Award.

References

[1] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. Scape: shape completion and animation of people. In *ACM transactions on graphics (TOG)*, volume 24, pages 408–416. ACM, 2005.

[2] S. Biasotti, A. Cerri, A. Bronstein, and M. Bronstein. Recent trends, applications, and perspectives in 3d shape similarity assessment. In *Computer Graphics Forum*, volume 35, pages 87–119. Wiley Online Library, 2016.

[3] F. Bogo, J. Romero, M. Loper, and M. J. Black. Faust: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014.

[4] J.-D. Boissonnat. Shape reconstruction from planar cross sections. *Computer vision, graphics, and image processing*, 44(1):1–29, 1988.

[5] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniah, and D. Dunaway. A 3d morphable model learnt from 10,000 faces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5543–5552, 2016.

[6] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 3189–3197, 2016.

[7] D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, and D. Cremers. Anisotropic diffusion descriptors. In *Computer Graphics Forum*, volume 35, pages 431–441. Wiley Online Library, 2016.

[8] G. Bouritsas, S. Bokhnyak, M. Bronstein, and S. Zafeiriou. Neural 3d morphable models: Spiral convolutional networks for 3d shape representation learning and generation. *arXiv preprint arXiv:1905.02876*, 2019.

[9] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[10] Z. Chen, X. Li, and J. Bruna. Supervised community detection with line graph neural networks. *arXiv preprint arXiv:1705.08415*, 2017.

[11] S. Cheng, I. Kotsia, M. Pantic, and S. Zafeiriou. 4dfab: A large scale 4d database for facial expression analysis and biometric applications. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5117–5126, 2018.

[12] N. Choma, F. Monti, L. Gerhardt, T. Palczewski, Z. Ronaghi, P. Prabhat, W. Bhimji, M. Bronstein, S. Klein, and J. Bruna. Graph neural networks for icecube signal classification. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 386–391. IEEE, 2018.

[13] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[14] M. Fey, J. Eric Lenssen, F. Weichert, and H. Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 869–877, 2018.

[15] P. Gainza, F. Sverrisson, F. Monti, E. Rodola, M. M. Bronstein, and B. E. Correia. Deciphering interaction fingerprints from protein molecular surfaces. *bioRxiv*, page 606202, 2019.

[16] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216. ACM Press/Addison-Wesley Publishing Co., 1997.

[17] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1263–1272. JMLR. org, 2017.

[18] V. G. Kim, Y. Lipman, and T. Funkhouser. Blended intrinsic maps. In *ACM Transactions on Graphics (TOG)*, volume 30, page 79. ACM, 2011.

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

[20] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[21] I. Kokkinos, M. M. Bronstein, R. Litman, and A. M. Bronstein. Intrinsic shape context descriptors for deformable shapes. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 159–166. IEEE, 2012.

[22] I. Kostrikov, Z. Jiang, D. Panozzo, D. Zorin, and J. Bruna. Surface networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2540–2548, 2018.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[24] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[25] I. Lim, A. Dielen, M. Campen, and L. Kobbelt. A simple approach to intrinsic correspondence learning on unstructured 3d meshes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[26] O. Litany, T. Remez, E. Rodola, A. Bronstein, and M. Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5659–5667, 2017.

[27] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 37–45, 2015.

[28] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5115–5124, 2017.

[29] F. Monti, M. Bronstein, and X. Bresson. Geometric matrix completion with recurrent multi-graph neural networks. In *Advances in Neural Information Processing Systems*, pages 3697–3707, 2017.

[30] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)*, 31(4):30, 2012.

[31] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black. Generating 3d faces using convolutional mesh autoencoders. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 704–720, 2018.

[32] O. Van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. In *Computer Graphics Forum*, volume 30, pages 1681–1707. Wiley Online Library, 2011.

[33] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

[34] N. Verma, E. Boyer, and J. Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2598–2606, 2018.

[35] K. Veselkov, G. Gonzalez, S. Aljifri, D. Galea, R. Mirnezami, J. Youssef, M. Bronstein, and I. Laponogov. Hyperfoods: Machine intelligent mapping of cancer-beating molecules in foods. *Scientific reports*, 9(1):9237, 2019.

[36] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. In *ACM Transactions on Graphics (TOG)*, volume 27, page 97. ACM, 2008.

[37] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.