Neural 3D Morphable Models: Spiral Convolutional Networks for 3D Shape Representation Learning and Generation Applied to the BU3DFE Dataset

Lorenzo Mandelli, Paula Mihalcea
Università degli Studi di Firenze
{lorenzo.mandelli, paula.mihalcea}@stud.unifi.it

Abstract

Generative neural models for 3D shapes are being used in an increasing number of applications, such as 3D computer vision and graphics, animation, game design and 3D printing. The generalization of successful 2D architectures, such as convolutional neural networks (CNNs), to arbitrary 3D geometries (graphs) remains however a partially unsolved problem, as though in spite of the many approaches that have been proposed, none possess all the desirable properties of such traditional models. In this paper we focus on the novel architecture introduced in [9], named Neural3DMM, which tries to create an improved 3D counterpart of the convolutional operator by defining a spiral convolutional operator that can be applied directly to 3D meshes. In [9] the authors demonstrate state-of-the-art results on a variety of 3D shape datasets compared to other graph convolutional operators. We replicate and confirm their work on the COMA dataset [3], and expand it by introducing the BU3DFE dataset [21]. We thus test said state-of-the-art architecture on an older, smaller dataset, in order to check its behaviour when less refined data is given as input, as well as when different dataset splits are used. We also modify the model so as to explore the possibility of using a novel weighted L1 loss, and discuss its advantages and disadvantages over a classic L1 loss. In the end, our work shows that Neural3DMM achieves decent results even in less-than-ideal conditions, even though they are not as good as those obtained in the tests presented by Bouritsas et al. [9].

1. Introduction

The automatic generation of 3D shapes is a growing necessity in computer vision, graphics, cinema, game design and 3D printing applications, as it allows for shorter production times and better geometric data analysis. The need for reliable generative models for 3D geometric data is thus a pressing matter, as there exist many implementations but none has all the properties of the well-known, robust 2D convolutional neural networks (CNNs).

The reason behind this problem is the impossibility of uniquely defining an analogous convolutional operator on data with non-Euclidean structure, like manifolds, graphs and, of course, 3D meshes; in fact, the most successful existing models actually consist of a series of different proposals for a sort of 3D convolution, each with its own pros and cons.

Many approaches use intermediate 3D data representations, such as voxels, point clouds, bidimensional images generated from 3D models and implicit-surfaces, but learning directly on the mesh allows to reduce the need for complex models and large collections of data. Nonetheless, this approach has been only recently explored [27, 3, 24, 25], and for this reason it is the one that we focus on in our work.

In this paper we work with a novel representation learning and generative architecture for fixed topology meshes which uses an ordering-based graph convolutional operator named *spiral convolution* by its authors [9]. Similarly to 2D image convolutions, the spiral convolutional operator defines an explicit ordering for each neighbor of each vertex on the mesh, an order obtained via a spiral scan, as proposed in [11]. This way anisotropic filters are obtained without sacrificing computational complexity while still explicitly encoding the fixed graph connectivity, and via the equivalence to 2D convolutions, common CNN practices can be easily adapted to meshes. The spiral convolution is used as a basic building block for the mesh autoencoders defined in [9], coined Neural 3D Morphable Models (Neural3DMM).

We quantitatively confirm its authors' state-of-the-art reconstruction results on the popular human faces dataset COMA [3], and proceed to add a new dataset to the list of the tested 3D data, namely the BU3DFE dataset [21]. We adapt these models to the format used by Neural3DMM and create two different splits, one by including a small percentage of each unique identity in the test set, and another by

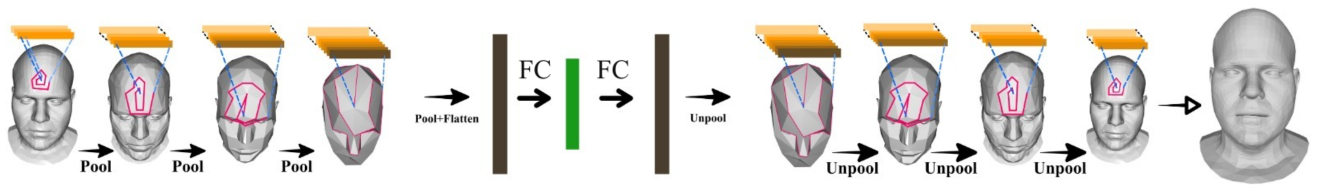

Figure 1: Illustration of the Neural3DMM architecture, from Bouritsas *et al.* [9]

completely excluding a number of identities from the train set. We perform various trainings with different latent vector dimensions in order to study how this hyperparameter influences the success of the model, and compare it to the results obtained with the COMA dataset by Bouritsas *et al.* [9].

We also modify the model in order to implement and test a novel weighted L1 loss [2], and discuss the results obtained by using it for the training of the network with our two dataset split protocols.

Finally, we show how the Neural3DMM model still behaves decently with a smaller dataset, as graphically seen in the reconstructed 3D faces, but commits larger error because of the limited data and its inferior quality.

2. Related work

Data representations: Among the most common approaches for arbitrary 3D shape generation we can find volumetric CNNs working on 3D **voxels** [38, 5, 14], which yield coarse and redundant representations in spite of their high computational complexity; **point clouds** [28, 4, 36], a simple and lightweight alternative to voxels, but which lack an underlying smooth structure; **image-based methods** [1, 32, 10], requiring pre- and post-processing steps and tending to produce undesirable artefacts; and finally, **implicit-surface** based approaches [37, 19, 13], which yield accurate results but with the disadvantage of slow inference. In contrast, as our work learns from **3D mesh** data, we are able to build invariance to both rigid and nonrigid shape transformations into the architecture, and thus reducing the amount of data and computations needed.

Geometric deep learning is another name for the generalization of common architectures to non-Euclidean data structures such as graphs and manifolds. Methods trying to achieve this goal have obtained promising results in areas like geometry processing and computer graphics, computational chemistry [6, 18] and network science [33, 8]. Many methods have been proposed to define operators similar to the convolution, among which there are spectral methods [33, 15, 23, 20] and soft attention mechanisms [30, 26].

Spiral convolution: As stated in section 1, one of the most challenging aspects of 3D shape generation is the difficulty of extending the convolutional operator to 3D spaces, and in particular the lack of a global system of coordinates that can be associated with each point of the mesh - a problem equivalent to the absence of a canonical ordering of the vertices.

Among the many approaches that have been devised there are the so-called *patch operator approaches*, like those in [16, 7, 8], which solve this problem by building a local system of coordinates around each vertex of the mesh, and then apply a series of weighting functions in order to aggregate information from the neighbouring vertices. These methods are somehow a generalization of the sliding window filtering in images; however, patch operator based approaches are insensitive to the graph topology, have great computational complexity, too many parameters, and allow for close to no optimization.

Other methods resembling our spiral convolution are those in [11] and [33, 23, 3]. However, Lim *et al.* [11] chose randomly the starting point of each spiral, resulting in the impossibility to encode the fixed connectivity of the graph - since corresponding vertices in different meshes do not undergo the same transformation, like in image convolutions. Spectral convolutional operators developed in [33, 23] and used by Ranjan *et al.* [3], on the other side, are inherently isotropic since they rely on the Laplacian operator, meaning that they are rather weak and not very expressive in comparison with our anisotropic spiral convolutional filters.

Morphable models are a kind of statistical model; in the case of deformable shapes, such as faces, a fixed topology can be obtained by establishing a dense correspondence with a given template model.

Let $\mathbf{F} = [\mathbf{f_0}|\mathbf{f_1}|..., \mathbf{f_N}]$, $\mathbf{f}_i \in \mathbb{R}^{d \cdot m}$ the matrix of all data defined on a set of meshes in dense correspondence that are sampled from a distribution $\mathcal{D}$, where d is the dimension of the data (vertex position, texture, color, etc.) and m the number of vertices. A linear 3D Morphable Model [34] represents arbitrary instances $\mathbf{y} \in \mathcal{D}$ as a linear combination of the k largest eigenvectors of the covariance matrix of $\mathbf{F}$ by assuming $\mathcal{D}$ to be a Gaussian distribution:

$$\mathbf{y} \approx \bar{\mathbf{f}} + \sum_{i}^{k} \alpha_i \sqrt{d_i} \mathbf{v_i}, \tag{1}$$

where $\bar{\mathbf{f}}$ is the mean shape, $\mathbf{v_i}$ the ith principal component, d_i the respective eigenvalue and α_i the linear weight coefficient. Because of its linear formulation, the actual representational power of the 3DMM in this form is constrained by its eigenvectors while its parameters scale linearly with respect to the number of the eigencomponents used, leading to large parametrisations for meshes of high resolution.

Facial expression generation guided by landmarks. Landmark detection from human faces is a reliable and accurate method [22, 35, 31, 17]. Landmarks are a valid way to account for facial deformation by focusing mostly on the face regions that are subject to stronger motions under facial expressions (e.g. those around the landmark points, like eye contour, nose, mouth, chin, etc.), while giving less importance to areas that remain relatively stable (like the forehead).

3. Spiral convolutional network

3.1. Spiral convolution

Suppose that we have a manifold, discretised as a triangular mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where $\mathcal{V} = \{1, ..., n\}$ is the set of vertices, $\mathcal{E}$ is the set of edges and $\mathcal{F}$ the set of faces. Let also $f : \mathcal{V} \to \mathbb{R}$ be a function representing the feature faces.

We note that any problems arising from the lack of a global ordering and insensitivity to graph topology are irrelevant when dealing with **fixed topology meshes**, like the ones used in our experiments. In particular, one can locally order the vertices and keep the order fixed by simply specifying a rotation direction (clockwise or counterclockwise) and a first, reference vertex.

Let us then define as $\{x_1, ..., x_L\}$ the set of neighboring vertices of each mesh vertex $x \in \mathcal{V}$, ordered in a fixed way, and $R^d(x)$ the *d-ring*, or an ordered set of vertices whose shortest graph path to x is exactly d hops long (subsequently $R^d_j(x)$ denotes the jth element in the d-ring, and $R^0_1(x) = x$). By defining the **spiral patch operator** as the ordered sequence:

$$S(x) = \{x, R^1_1(x), R^1_2(x), ..., R^h_{|R^h|}\}, \tag{2}$$

where h denotes the *patch radius* (a 3D equivalent of the kernel size in classical CNNs), then **spiral convolution** is:

$$(f * g)_x = \sum_{\ell=1}^{\infty} g_\ell f(S_\ell(x)) = 1. \tag{3}$$

As mentioned, the uniqueness of the ordering is given by fixing the direction of the rings (clockwise or counterclockwise) and the **reference point** $R^1_1(x)$, as the rest of the vertices of the spiral can be ordered inductively. In particular, given a fixed reference vertex x_0, we choose the initial point for each spiral to be in the direction of the shortest geodesic path to x_0, i.e.

$$R^1_1(x) = \operatorname*{argmin}_{y \in R^1(x)} d_{\mathcal{M}}(x_0, y), \tag{4}$$

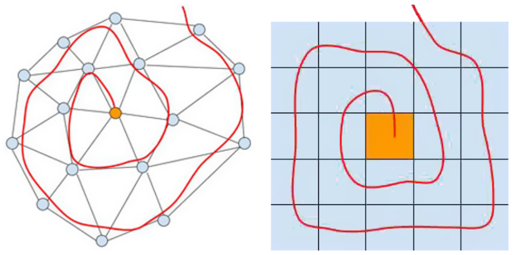

Figure 2: Spiral ordering on a mesh (left) and an image patch (right).

where $d_{\mathcal{M}}$ is the shortest geodesic distance between two vertices on the mesh $\mathcal{M}$. Furthermore, in order to allow for fixed-sized spirals, we choose a fixed length L as a hyper-parameter and then either truncate or zero-pad each spiral, depending on its size with respect to L.

3.2. Neural 3D morphable models

In order to overcome the linearity issues that come with the morphable model from equation 1, Bouritsas *et al.* [9] use spiral convolutions as a building block to construct a fully differentiable non-linear morphable model, in essence creating a deep convolutional mesh autoencoder that locally processes each shape and learns multi-scale representations (as shown in figure 1). This way the model also learns semantically meaningful representations of the input data, while bypassing the need to make assumptions about its distribution and greatly reducing the number of parameters.

In practice, similarly to traditional convolutional autoencoders, the model uses a series of convolutional layers with small receptive fields (equivalent to kernels) followed by pooling and unpooling layers, for the encoder and the decoder respectively, where a decimated or upsampled version of the mesh is obtained each time and the features of the existing vertices are either aggregated or extrapolated. The calculation of the features after upsampling is done according to [3], i.e. through interpolation by weighting the nearby vertices.

3.3. Weighted loss function

Having established that all meshes have a fixed topology and are in full point-to-point (dense) correspondence, let us introduce a new notation where $\mathcal{L} = \{(\mathbf{S}^n_1, \mathbf{S}^e_1, Z^e_1), ..., (\mathbf{S}^n_m, \mathbf{S}^e_m, Z^e_m)\}$ is the training set, with $\mathbf{S}^n_i = (p^n_1, ..., p^n_N) \in \mathbb{R}^{N\times3}$ a neutral 3D face, $\mathbf{S}^e_i = (p^e_1, ..., p^e_N) \in \mathbb{R}^{N\times3}$ an expressive 3D face and $Z^e_i \in \mathbb{R}^{n\times3}$ the 3D landmarks corresponding to $\mathbf{S}^e_i$. N is the number of total vertices of each mesh, n the number of landmarks and m the number of examples in the training set.

The original Neural3DMM model from Bouritsas *et al.* [9] was trained towards self-reconstruction, which corre-

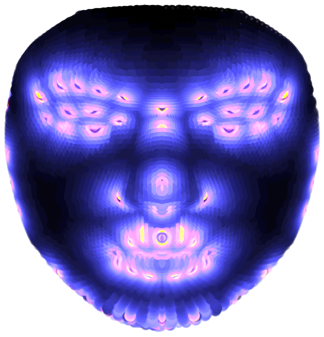

Figure 3: Distribution of the weights for the BU3DFE template mesh, computed from its 66 landmarks; darker areas correspond to weights close to zero, although they are not all equal and for computational reasons none of them are really null.

sponds to finding a function $h : \mathbb{R}^{N\times3} \rightarrow \mathbb{R}^{N\times3}$ such that $\mathbf{S}^g \approx h(\mathbf{S}^{in})$. This was achieved by minimizing the **L1 loss**, computed between the input mesh $\mathbf{S}^{in}$ and the generated output $\mathbf{S}^g$.

Differently, as proposed in [2], we have modified the loss function in order to give more importance to the movable regions of the face, so as to generate a novel 3D face $\mathbf{S}^g$ by focusing on reproducing the expression provided by the target landmarks Z^e while yet maintaining the identity structure of $\mathbf{S}^n$. We thus implemented the **weighted L1 loss** introduced in [2]:

$$L(\mathbf{S}^e, \mathbf{S}^g) = \frac{1}{N}\sum_{i=1}^{N} w_i \cdot \parallel p_i^e - p_i^g \parallel_1 . \qquad (5)$$

The authors of [2] define the weights as the inverse of the Euclidean distance of each vertex p_i in the mesh from its closest landmark Z_j. This provides a coarse indication of the location of each vertex, and how much it contributes to the expression generation. Given that the mesh topology is fixed, we pre-compute the weights w_i and then reuse them for each sample.

For each vertex p_i on the template mesh, we compute the weight w_i as:

$$w_i = \frac{1}{\min d(p_i, Z_j)} \forall j, \qquad (6)$$

where $d(\cdot)$ is the Euclidean distance; weights are then normalized so that they lie in the $[0; 1]$ range. Vertices corresponding to the landmarks, i.e. $p_i = Z_j$ for some j, are hence assigned the maximum weight.

4. Experiments

In this section we show the response of Bouritsas *et al.*'s Neural3DMM network [9] in the presence of non-ideal dataset conditions and by varying the size of the latent vector, as well as by using the novel loss function introduced by [2], the weighted L1 loss.

First, we test the model on the COMA dataset, in order to verify the results presented in [9] and obtain the data needed to enable a comparison with BU3DFE, the newly used dataset.

We continue by performing a series of experiments with different split protocols of the aforementioned BU3DFE dataset, so as to test the generative capabilities of the network and in particular its ability to predict expressions of faces not previously seen during the training phase.

Furthermore we explore the model's performance by varying the dimension of the latent vector, whose value proved critical to improve the results generated by the Neural3DMM architecture, as well as the training loss function.

Finally, we discuss the results obtained from our experiments and compare them with those in [9], and analyze the possible benefits of introducing a weighted loss for the 3D mesh generation of deformable shapes.

All code is freely available at https://github.com/PaulaMihalcea/Neural3DMM_BU3DFE.

4.1. Datasets and experimental setting

COMA. The facial expressions dataset from Ranjan *et al.* [3] contains 20,466 $\mathbf{S} \in \mathbb{R}^{N\times3}$ meshes ($N = 5,023$ vertices) of 12 extreme, asymmetric expressions captured over 12 different subjects. This is actually a common benchmark employed in other studies ([3, 9]).

BU3DFE. The dataset from Yin *et al.* [21] contains 1,779 $\mathbf{S} \in \mathbb{R}^{N\times3}$ 3D scans ($N = 6,704$ vertices) of 100 unique identities (56% female, 44% male), of age ranging from 18 to 70 years old, and with a variety of ethnic/racial ancestries (including White, Black, East-Asian, Middle-east Asian, Indian, and Hispanic Latino). The subjects perform seven different facial expressions, distributed as shown in table 1.

Expression	No. of models	% of dataset
Neutral	57	3.2
Angry	158	8.88
Disgust	351	19.73
Fear	349	19.62
Happy	347	19.51
Sad	161	9.05
Surprise	356	20.01

Table 1: Expressions distribution in the BU3DFE dataset.

Split protocols. For the training of the Neural3DMM model we used two different split protocols, which we named ***percentage split*** and ***identity split***. According to the first of these protocols, we train the model on a fixed percentage of all models of each identity, so as to allow it to see at least one model of each person in the dataset, then test it on the remaining models (which also contain at least one model per identity); this is also the split protocol originally used by Bouritsas *et al.* [9], and is characterized by a lower difference between the elements in the training set and those in the test set. The second protocol uses an inverse approach instead, as it reserves a certain number of identities for the test split and makes sure that none of their models are seen by the network during training - so as to verify the generalization to unseen identities.

4.2. Implementation details

For the network architecture we used the structure used by Bouritsas *et al.* [9] for COMA, given the similarities of this dataset with BU3DFE.

Thus we denote with $SC(h, w)$ a spiral convolution of h hops and w filters, $DS(p)$ and $US(p)$ a downsampling and an upsampling by a factor of p respectively, $FC(d)$ a fully connected layer, d the latent vector size and l the number of vertices after the last downsampling layer. The resulting network architecture then is as follows:

Encoder: $SC(1,16) \rightarrow DS(4) \rightarrow SC(1,16) \rightarrow DS(4) \rightarrow SC(1,16) \rightarrow DS(4) \rightarrow SC(1,32) \rightarrow DS(4) \rightarrow FC(d)$;

Decoder: $C(l * 32) \rightarrow US(4) \rightarrow SC(1,32) \rightarrow US(4) \rightarrow SC(1,16) \rightarrow US(4) \rightarrow SC(1,16) \rightarrow US(4) \rightarrow SC(1,3)$.

All of the activation functions were ELUs (exponential linear unit functions, see eq. 7), a differentiable non-saturating (thus reliable) alternative to RELUs, despite its slightly higher computation time. The network used a learning rate of 10^{-3} with a decay of 0.99 after each epoch and a weight decay of $5 \cdot 10^{-5}$; all models were trained for 300 epochs.

$$R(z) = \begin{cases} z & \text{for } z > 0 \\ \alpha \cdot (e^z - 1) & \text{for } z \leq 0 \end{cases}. \tag{7}$$

4.3. Evaluation

In order to obtain quantitative results and enable a direct comparison with the experiments of Bouritsas *et al.* [9], we used the same evaluation metric, namely the **generalization error**, which measures the ability of a model to represent novel shapes from the same distribution as it was trained on. More specifically we evaluate the average per sample and per vertex euclidean distance in the 3D space (in millimetres) between corresponding vertices in the input and its reconstruction.

We also rendered some of the test meshes in their original and reconstructed form, applying heatmaps to the latter, so as to allow for a qualitative evaluation of the results, too.

4.4. Results

4.4.1 Comparison to COMA dataset

The first tests aimed to confirm the results for the COMA dataset presented by Bouritsas *et al.* in [9], which were made using a percentage split where 90% of the models of each identity were contained in the training set, while the remaining 10% models were left for the test set.

After having effectively reproduced their findings, we proceeded by training the network on the BU3DFE dataset, using the same split. We found appropriate to also keep the model configuration, given the two datasets' similarities. Indeed, in spite of the larger number of examples, the COMA dataset actually is way less diversified than BU3DFE, since it only contains 12 distinct identities (versus the 100 identities of BU3DFE) and is characterized by a series of temporal scans of the persons performing different expressions, resulting in an overall redundancy which the BU3DFE dataset does not have, regardless of its 1,779 models (versus the 20,466 examples of COMA).

We actually think this to be reason why, as seen in figure 4, the improvement of the generalization error on the COMA dataset is not as great as the latent vector increases, while the error on BU3DFE keeps decreasing significantly as this dimension becomes larger; we also hypothesize that the different topologies characterizing the meshes of these two datasets might have influenced this result.

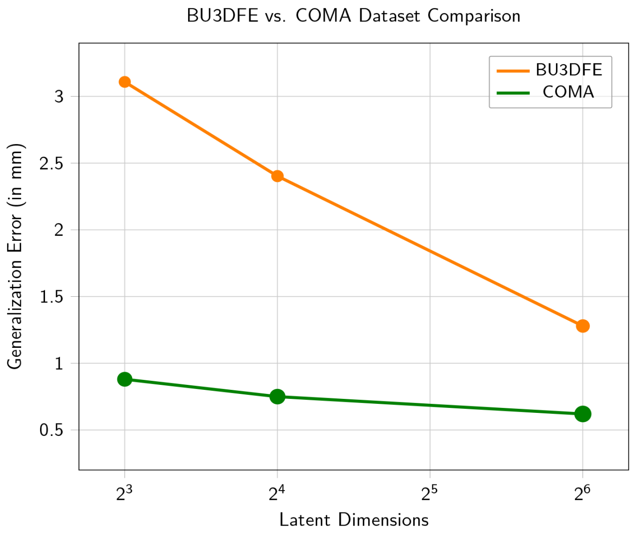

Figure 4: Quantitative evaluation of the performance of the BU3DFE dataset vs COMA, in terms of generalization error.

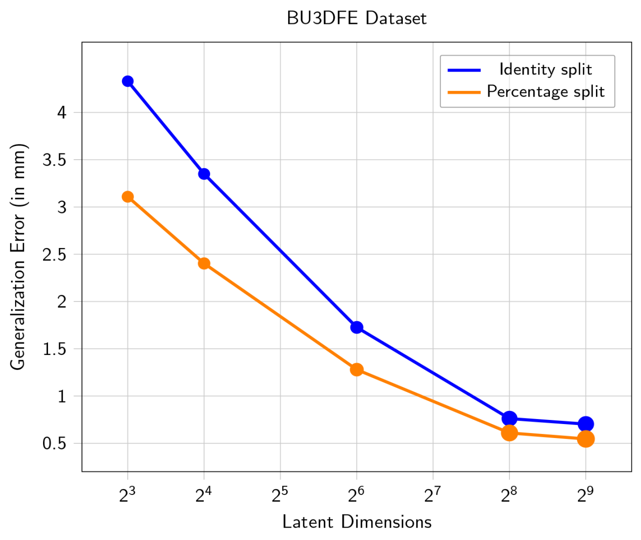

Figure 5: Performance of an identity vs a percentage split of the BU3DFE dataset, at varying latent dimensions.

4.4.2 Dataset split comparison

In order to test the robustness of the network, we proceeded by using two different protocols for dividing the models in the BU3DFE dataset into the training and test sets, the **percentage split** and the **identity split**. In each of them, the training set contained 90% of all models for each identity and 90% of all identities (or equivalently, all models of 92 identities), respectively.

As expected, the percentage split achieved better results, as the evaluation was conducted on a test set containing identities which had already been seen during the training phase (since it included at least one model for each of the 100 people in the dataset).

Inversely, when using the identity split the model had more trouble reconstructing never-before-seen faces, although it is worth noting that it almost closed the gap with the percentage split at higher latent vector dimensions (specifically, latent dimensions over 256), as observed in figures 5 and 6 - to the point that we could consider using a latent vector of 256 instead of 512, in order to save training time and computational complexity (as it would greatly reduce the number of model parameters).

4.4.3 Weighted loss comparison

After having previously observed that a latent dimension equal to 512 yields the best results, we kept it fixed and continued our experiments by implementing the weighted loss function defined in section 3.3 and then performing two trainings, one for each dataset split protocol.

In spite of our expectations, we observe that this strategy does not provide a significant improvement with respect to the standard L1 loss, although further experiments with different weights distributions could be performed in order to verify this hypothesis. Figure 6 illustrates the cumulative per-vertex Euclidean error of these studies, and shows that overall the weighted L1 loss is not a very convenient loss neither in terms of computational complexity nor parameters-to-performance ratio, meaning that an equiva-

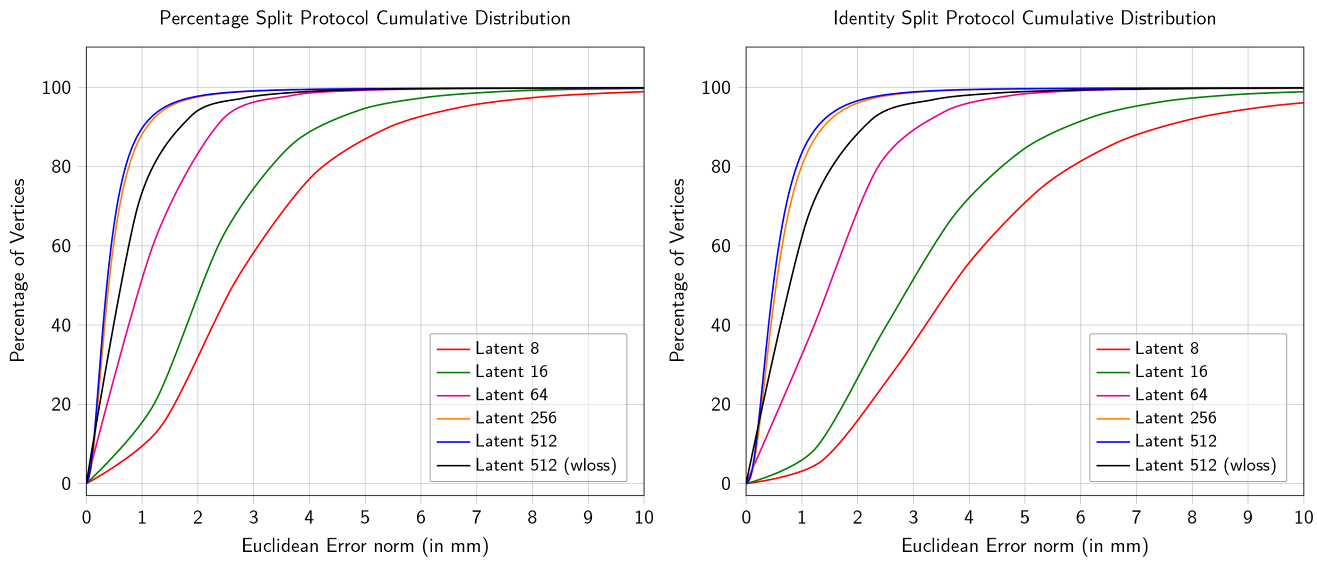

Figure 6: Cumulative per-vertex Euclidean error between different latent dimensions for the BU3DFE dataset percentage split, including that of the weighted loss trainings.

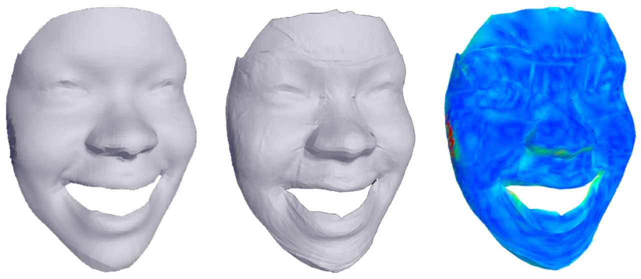

Figure 7: Reference (left) and reconstructed meshes (middle), with the corresponding color heatmap of the error (right), for the percentage split with latent vector 512.

lent training using a classic L1 loss with latent vector 512 or 256 achieves better results in less time.

4.4.4 Qualitative evaluation

In this section we determine the representational capacity of the Neural3DMM model when confronted with a different dataset and its different splits by graphically comparing the reconstructed meshes to their original counterparts.

Percentage split. Figure 7 shows some of the reconstructed meshes for this dataset split with latent vector equal to 512. We observe that they are qualitatively good, and resemble rather well their respective ground truths, even though they do not reach the realism levels displayed by Bouritsas *et al.*'s work in [9].

Identity split. We analyze here the predicted faces for the identity split. We note that, in accordance with the results previously obtained in terms of generalization error, we got less refined reconstructions, even at larger latent dimensions. Nonetheless, even more extreme expressions remain well recognizable (see figure 8), although visibly noisier than those generated with the percentage split training.

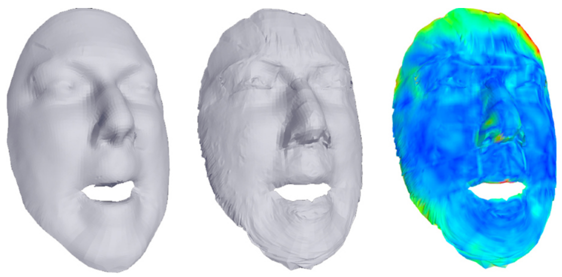

Figure 8: Reference (left) and reconstructed meshes (middle), with the corresponding color heatmap of the error (right), for the identity split with latent vector 512.

Weighted loss. These meshes were those for which we hoped to see the best results in the most extreme expressions, and although they do not meet all our expectations, they do not disappoint either. We observe in figure 9 how they achieve good-looking reconstruction, especially in the percentage split case (top row) and even for extreme expressions, on a level comparable to that of the meshes obtained from the classic L1 loss - despite the overall greater generalization error seen in fig. 6. Furthemore, the areas with the largest concentration of landmark points look, qualitatively speaking, more refined than the rest of the face, particularly the nose and the eye contour, proving that weighted points meant to focus the network on the most mobile parts of the mesh at least partially does their job, even though at a first glance it seemed otherwise.

5. Conclusions

In this paper we described and analyzed a state-of-the-art architecture for fixed topology 3D deformable shapes which uses a novel mesh convolutional operator, the spiral convolutional operator.

We began proving some of the results obtained by its original authors and proceeded with the introduction of a different dataset, the BU3DFE dataset, which has never been used together with the examined architecture. We defined two different split protocols, one of which had not been tested before, and used them to train the network with

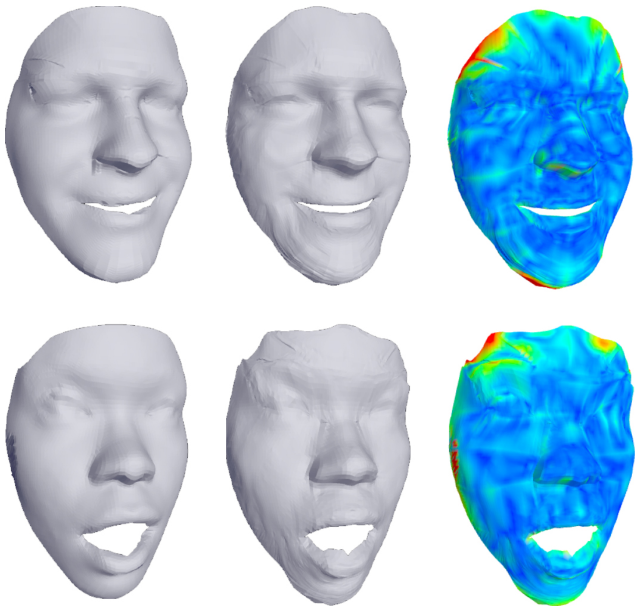

Figure 9: Reference (left) and reconstructed meshes (middle), with the corresponding color heatmap of the error (right), for the weighted loss training with percentage split (top row) and identity split (bottom row), with latent vector 512.

different latent vector sizes. We showed to which point the latent dimension, combined with a given dataset split, affects the generalization error on the test set, and compared it with the original paper's results.

We also modified the original architecture in order to implement a novel weighted L1 loss, then used it to train our network and showcase the effects of an approach which focuses on the most mobile parts of the human face through the use of landmark points. Although we did not obtain outstanding results, we still got visibly better reconstructions of the areas closer to the aforementioned landmarks.

In essence, our work showed that the Neural3DMM architecture achieves decent results even in less-than-ideal conditions, with small datasets and different split protocols; the size of the latent vector proved to be fundamental in order to obtain better results, and the introduction of a weighted L1 loss yielded an outcome which suggests that further experiments may be performed in this regard.

Additionally, during our work on this paper we had to solve a series of issues regarding the implementation of Ranjan *et al.* and Bouritsas *et al.*'s papers [3, 9], which ultimately led us to contribute to one of the main libraries used by them, namely the Max Planck Intelligent Systems PSBody Mesh package, or **MPI-IS PSBody Mesh package** [12]. Our addition consists of a relevant extension to the documentation which allows the package to be installed on a wider range of machines; after intensive testing and debugging, this addendum has been officially approved [29] and will most likely be included in the library's next release.

6. Acknowledgments

This work has been conducted under the exceptional guide and supervision of Claudio Ferrari, Ph.D, as part of the Visual and Multimedia Recognition course held by Prof. Alberto Del Bimbo and Federico Becattini, Ph.D, from the Media Integration and Communication Center (MICC) at Università degli Studi di Firenze.

References

[1] Amir Arsalan Soltani, Haibin Huang, Jiajun Wu, Tejas D Kulkarni, and Joshua B Tenenbaum. Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[2] Anonymous ICCV submission. 3d to 4d facial expressions generation guided by landmarks. *ICCV 2021 Submission #9955*, 2021. 2, 4

[3] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3d faces using convolutional mesh autoencoders. *European Conference on Computer Vision (ECCV)*, 2018. 1, 2, 3, 4, 8

[4] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[5] t. . V. Daniel Maturana and Sebastian Scherer. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015. 2

[6] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in Neural Information Processing systems (NIPS)*, 2015. 2

[7] Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[8] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[9] Giorgios Bouritsas, Sergiy Bokhnyak, Stylianos Ploumpis, Michael Bronstein, and Stefanos Zafeiriou. Neural 3d morphable models: Spiral convolutional networks for 3d shape representation learning and generation. *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 1, 2, 3, 4, 5, 7, 8

[10] Heli Ben-Hamu, Haggai Maron, Itay Kezurer, Gal Avineri, and Yaron Lipman. Multi-chart generative surface modeling. *SIGGRAPH Asia 2018 Technical Papers, page 215. ACM*, 2018. 2

[11] Isaak Lim, Alexander Dielen, Marcel Campen, and Leif Kobbelt. A simple approach to intrinsic correspondence learning on unstructured 3d meshes. *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)*, 2018. 1, 2

[12] Jean-Claude Passy, Anurag Ranjan, Timo Bolkart, Ivan Oreshnikov, Kenneth Chaney, Dávid Komorowicz. MPI-IS PSBody Mesh package. 2018. https://github.com/MPI-IS/mesh. 8

[13] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *CVPR*, 2019. 2

[14] JiajunWu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[15] Joan Bruna,Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *International Conference on Learning Representations (ICLR)*, 2014. 2

[16] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. *Proceedings of the IEEE In-*

ternational Conference on Computer Vision Workshops (ICCVW), pages 37–45, 2015. 2

[17] JunWan, Zhihui Lai, Jing Li, Jie Zhou, and Can Gao. Robust facial landmark detection by multiorder multiconstraint deep networks. *IEEE Trans. on Neural Networks and Learning Systems, pages 1–14*, 2021. 3

[18] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017. 2

[19] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. *CVPR*, 2019. 2

[20] Li Yi, Hao Su, Xingwen Guo, and Leonidas J Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[21] Lijun Yin, Xiaozhou Wei, Yi Sun, Jun Wang, and Matthew J Rosato. A 3d facial expression database for facial behavior research. *7th International Conference on Automatic Face and Gesture Recognition*, 2006. 1, 4

[22] Lisha Chen, Hui Su, and Qiang Ji. Deep structured prediction for facial landmark detection. *Advances in Neural Information Processing Systems (Neurips), volume 32*, 2019. 3

[23] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[24] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. *ECCV*, 2018. 1

[25] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. *CVPR*, 2019. 1

[26] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR), pages 2598–2606*, 2018. 2

[27] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1886–1895*, 2018. 1

[28] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. *International Conference on Machine Learning (ICML)*, 2018. 2

[29] Paula Mihalcea. Conda installation guide for the MPI-IS PSBody Mesh package. 2021. `https://github.com/MPI-IS/mesh/pull/58`. 8

[30] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *International Conference on Learning Representations (ICLR)*, 2018. 2

[31] Rongye Meng, Sanping Zhou, Xingyu Wan, M. Li, and Jinjun Wang. Teacher-student asynchronous learning with multi-source consistency for facial landmark detection. *ArXiv, abs/2012.06711*, 2020. 3

[32] Stylianos Moschoglou, Stylianos Ploumpis, Mihalis Nicolaou, and Stefanos Zafeiriou. 3dfacegan: Adversarial nets for 3d face representation, generation, and translation. *arXiv preprint arXiv:1905.00307*, 2019. 2

[33] t. . S. Thomas N Kipf and Max Welling. *International Conference on Learning Representations (ICLR)*, 2017. 2

[34] Volker Blanz, Thomas Vetter, et al. A morphable model for the synthesis of 3d faces. *SIGGRAPH*, 1999. 2

[35] Xuanyi Dong, Yi Yang, Shih-En Wei, Xinshuo Weng, Yaser Sheikh, and Shoou-I Yu. Supervision by registration and triangulation for landmark detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence, pages 1–1*, 2020. 3

[36] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[37] t. . L. Zhiqin Chen and Hao Zhang. *CVPR*, 2019. 2

[38] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2