Proceedings of Machine Learning Research 148:87-99, 2021 NeurIPS 2020 Preregistration Workshop

PCA Retargeting: Encoding Linear Shape Models as Convolutional Mesh Autoencoders

Eimear O' Sullivan E.O-SULLIVAN16@IMPERIAL.AC.UK
Stefanos Zafeiriou S.ZAFEIRIOU@IMPERIAL.AC.UK
Imperial College London, UK

Abstract

3D Morphable Models have long played a key role in the construction of statistical shape models. While earlier models employed Principal Component Analysis, recent work has migrated towards mesh autoencoder models for the construction of lightweight, non-linear shape models that facilitate state-of-the-art reconstruction and the capture of high-fidelity details. Doing so results in a loss of interpretability and regularisation in the model latent space. To address this, we propose *PCA retargeting*, a method for expressing linear PCA models as mesh autoencoders and thereby retaining the gaussianity of the latent space. To encourage the capture of mesh details outside the expressive range of a PCA model, we introduce "free" latent space parameters. Experiments demonstrate the successful retargeting of the PCA models as mesh autoencoders. The introduction of "free" latent parameters have a greater impact when smaller latent vector sizes are used, but do not lead to any gains in reconstruction fidelity.

1. Introduction

3D Morphable Models (3DMMs) are one of the foremost technologies for the statistical analysis and modelling of the human face and body. Earlier examples typically employed principal component analysis (PCA) for model construction Blanz and Vetter (Jul 1, 1999), however recent years have seen the successful application of geometric deep learning leading to impressive gains in model reconstruction accuracy and representational power. Many of these gains comes from the introduction of convolutional mesh autoencoders Ranjan et al. (2018) and increased efficiency of mesh convolution operators Bouritsas et al. (2019); Defferrard et al. (2016); Fey et al. (2018); Gong et al. (2019).

As with standard autoencoder models, convolutional mesh autoencoders are subject to the loss of linear independence and regularity in the latent space that is inherent in PCA models. An example of where this might be problematic would be an attempt to create a blendshape model using a mesh autoencoder network. A blendshape model can be considered as a weighted sum of target expressions that can be combined with a neutral mesh to achieve facial animation Cao et al. (2014); Cheng et al. (2018). The linear independence of distinct blendshapes is paramount to their success. As autoencoder models do not offer any control over the latent space distribution, this linear independence cannot be preserved and autoencoder models of the human face typically address expressions in isolation Ranjan et al. (2018).

In this paper we propose the concept of *PCA retargeting*, a means of converting a PCA model to a mesh autoencoder format. Our aim is to train an autoencoder model to accurately reconstruct a target mesh while constraining the latent space to mimic the shape parameters the target mesh would produce in a PCA model. Autoencoders are typically trained end-to-end, where the reconstruction loss is calculated based on the output of the decoder and backpropagated through the network. This encourages the model to find an optimal encoding in the bottleneck layer that will permit the decoder to accurately reconstruct an input from its latent representation. To faithfully retarget a PCA model as an autoencoder, however, the latent space of the autoencoder model must follow the PCA shape space as precisely as possible. To enforce this constraint on the latent space, we draw inspiration from variational autoencoders (VAEs). VAEs can be considered autoencoders whose encoding space has been regularised during training to ensure desirable properties for data generation Kingma and Welling (2014); Rezende et al. (2014). A Gaussian distribution is often imposed on the latent space, though von Mises-Fisher Davidson et al. (2018); Xu and Durrett (2018) and Dirichlet distributions Joo et al. (2020) have also proven effective. This desired latent distribution is achieved by adding a regularisation term to the loss backpropagated through the model. By imposing a loss on the model latent space, we aim to compel the encoder to learn the desired latent representation.

The representational power of a PCA model is constrained by the number of linear eigenvectors it contains, and large models are required to achieve high resolution meshes Booth et al. (2018). To allow the retargeted model to represent high-fidelity mesh details, we propose to extend the length of the autoencoder latent vector beyond the number of shape parameters used in the retargeting process. As these additional parameters will not be constrained to follow the PCA shape parameters, the intention is that they will model details that are not expressed within the shape variation of the PCA model. We refer to these additional parameters as "free" latent variables.

Our analysis will address two main questions; 1) Can a PCA model be retargeted as a mesh autoencoder? 2) Can "free" latent parameters be used as an inexpensive means of capturing high-fidelity details? Experiments will be conducted using a large-scale facial dataset, and latent representations of varying sizes will be explored. We hypothesise that imposing rigid constraints on the latent space will allow for the training of an autoencoder model that operates as a neural analogue to a PCA model and that the introduction of "free" latent variables will permit the encoding of high frequency mesh details that can be lost when later PCA model eigencomponents are omitted.

2. Related Work

3D Morphable Models (3DMMs) facilitate the continuous parameterisation of shape variation for a given object class by performing dimensionality reduction on training dataset of meshes Blanz and Vetter (Jul 1, 1999). Gaussian Processes Lüthi et al. (2018) and linear blend-skinning Loper et al. (2015); Romero et al. (2017) have both been used for the construction of 3DMMs, though Principal Component Analysis (PCA) is perhaps the most prevalent approach Blanz and Vetter (Jul 1, 1999); Booth et al. (2018); Dai et al. (2017); Lüthi et al. (2018); Ploumpis et al. (2020). Let X be a set of n densely registered meshes, $\{x_1, x_2, ..., x_n\}$, sampled from a given distribution $\mathcal{D}$, where each mesh has d vertices con-

nected in a fixed topology. By making a gaussianity assumption, an arbitrary instance, $x^* \in \mathcal{D}$, can then be represented as a linear combination of the k largest eigenvectors of the covariance matrix of X:

$$x^* \approx \bar{x} + \sum_{i=1}^{k} \alpha_i U_i \tag{1}$$

where $\bar{x}$ is the mean shape, U_i is the i^{th} shape eigenvector, and α_i is the corresponding eigenvalue. Given the linear model representation, the largest shape variations are captured within the first eigenvectors. High resolution models therefore require a larger number of eigenvectors to be retained, while high finer details can be omitted when later principal components are discarded.

Advances in the field of geometric deep learning have led to the generalisation of neural networks to non-Euclidean domains, such as graphs and manifolds Bronstein et al. (2017). These emerging techniques have been applied across multiple domains, including computer graphics Boscaini et al. (2016); Masci et al. (2015), molecular prediction Veselkov et al. (2019), node classification Kipf and Welling (2016), and social network analysis Monti et al. (2019); Tan et al. (2019). Many approaches have been applied to extend standard 2D convolutions to non-Euclidean domains, including spectral methods Bruna et al. (2014); Fey et al. (2018); Levie et al. (2019); Tang et al. (2019), local charting based approaches Boscaini et al. (2016); Lim et al. (2018); Masci et al. (2015), and convolution operators that act directly on 3D mesh topology Bouritsas et al. (2019); Gong et al. (2019). New methods for graph coarsening have also been developed Diehl et al. (2019); Ying et al. (2018), while advances in mesh pooling and unpooling operations have led to the introduction of convolutional mesh autoencoders, an effective means of modelling 3D data Ranjan et al. (2018).

Feature disentanglement permits an intuitive understanding of the latent space of deep generative models. Supervised methods typically require a-priori knowledge of the nature and quantity of generative factors Hinton et al. (2011); Kulkarni et al. (2015); Reed et al. (2014). Higgins *et.al.* presented β-VAE, a reformulation of the standard VAE framework that allowed for a disentangled representation of generative data factors by introducing a β coefficient Higgins et al. (2017). Mathieu *et al.* argue that while β-VAE allows for control over the extent of overlap between latent data encodings, it does little to ensure that the aggregate data encodings conform to a desired structure. Instead, they present an approach that permits a desired latent encoding to be achieved by careful choice when defining the prior Mathieu et al. (2019). GANs Chen et al. (2016), Wasserstein Autoencoders Gaujac et al. (2020), different interpretations of VAEs Kulkarni et al. (2015); Shao et al. (2020), and mutual information maximisation Bachman et al. (2019); Hjelm et al. (2019); Tschannen et al. (2020) have also been applied successfully for feature disentanglement. In this work, as the desired latent encoding is known in advance, this problem is explored from a supervised angle.

3. Model Architecture and Training

Our mesh autoencoder model follows the architecture design of Bouritsas et al. (2019). The encoder is comprised of five mesh convolutional and pooling layers. The decoder architecture

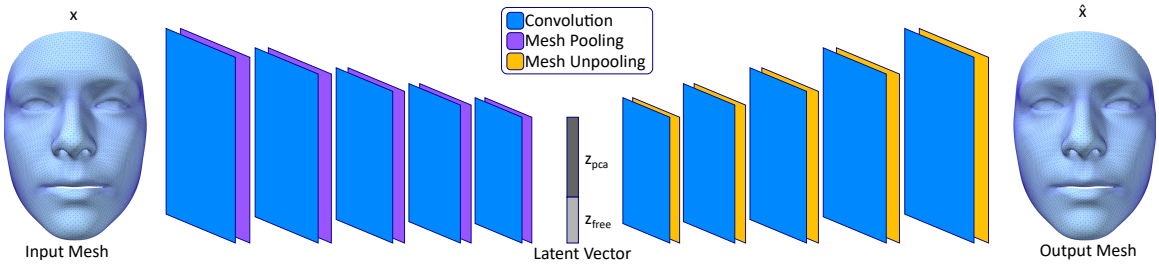

Figure 1: Network Architecture.

is the reverse of the encoder; five convolutional layers, each followed by a mesh unpooling layer. To facilitate the mesh pooling and unpooling operations, we follow the approach of Ranjan et al. (2018). In each pooling layer the number of mesh vertices will be reduced by a factor of 4. Spiral convolutions with a fixed length of 9 will be used in all convolution layers Gong et al. (2019). The complete network architecture is shown in Figure 1.

The framework will be implemented in PyTorch Paszke et al. (2017). Model hyperparameters and loss weights will be determined using a grid-search strategy.

3.1. Losses

For a given input mesh, x, an L1 reconstruction loss will be applied to the decoder output, $\hat{x}$. To encourage the latent vector representation to conform to the desired encoding, z_{pca}, we add a regularisation term to the loss function. In standard VAE models, the regularisation terms typically manifests as the Kullback-Leibler (KL) divergence between the desired and achieved encoding distributions. Here, the desired latent encoding follows a standard normal distribution by design. As such, the KL regularisation term is replaced by an L1 loss on the difference between the desired latent encoding, z_{pca}, and the achieved encoding, $\hat{z}_{pca}$. Defining λ_{rec} and λ_{reg} as the weights applied to the reconstruction and regularisation loss, respectively, the updated loss function takes the form:

$$loss = \lambda_{rec}||x - \hat{x}|| + \lambda_{reg}||z_{pca} - \hat{z}_{pca}|| \tag{2}$$

3.2. "Free" Latent Variables

In addition to the latent vector components reserved to correspond with the principal components of the PCA model, we also propose to extend the latent vector size by including l additional variables. As discussed in the Introduction, PCA models capture the modes of greatest variation in first few principal components while the later components are reserved for the higher frequency details. These later components are often omitted for reasons of model size and computational speed, and this can lead to a "smoothing" of the mesh surface. By adding l "free" variables to the latent vector, we aim to capture these higher frequency details while preserving the advantage of independence among the model components provided by the orthogonal bases of the PCA models.

To conform with the distribution of the other latent variables, we would prefer that the distribution of additional "free" latent variables also take the form of a standard normal distribution with zero mean and unit variance. A KL divergence will be used to encourage

the "free" variables to follow the desired Gaussian distribution. Defining the PCA latent variables as z_{pca} and the "free" latent variables as z_{free}, the complete loss function will take the form:

$$loss = \lambda_{rec}||x - \hat{x}|| + \lambda_{reg}||z_{pca} - \hat{z}_{pca}|| + \lambda_{reg}KL[\mathcal{N}(\mu_{z_{free}}, \sigma_{z_{free}}), \mathcal{N}(0, 1)] \quad (3)$$

where $\mu_{z_{free}}$ and $\sigma_{z_{free}}$, the mean and standard deviation of the "free" latent variables, will be calculated per batch during training. The full length of the latent vector is now given as the combined length of z_{pca} and z_{free}. For the "free" latent variables, no assumption of independence between variables will be made, as this criteria is not enforced by the proposed loss function.

4. Experimental Protocol

Tests will be conducted using latent vectors of size 16, 32, 64, and 128. For each latent vector size, three models will be trained and the mean and standard deviation results will be reported. Autoencoder models will be compared to baseline PCA models with the corresponding number of principal components. The number of model parameters will be recorded in each case.

4.1. Dataset

Models will be evaluated using the MeIn3D facial database Booth et al. (2018) which consists of more than $10,000$ scans registered to a template with $28,431$ vertices. Data will be split into $9k$ training and $1k$ testing meshes. Random stratified sampling will be applied to maintain gender, age, and ethnic proportions.

To obtain the desired latent vector encoding for all meshes, a PCA model will be constructed from the training dataset. The first 128 principal components will be retained. Shape parameter vectors, α, for each instance will be obtained by projecting the mesh instance onto the model. Following the probabilistic interpretation of PCA models, each parameter, α_i, is an independent random variable. Each α_i follows a Gaussian distribution with zero mean and a variance of λ_i, where $32\lambda_i$ is the i-th PCA eigenvalue. A shape vector with k components can therefore be normalised as follows:

$$z_{pca} = \frac{\alpha_1}{\sqrt{\lambda_1}}, \frac{\alpha_2}{\sqrt{\lambda_2}}, \cdots, \frac{\alpha_k}{\sqrt{\lambda_k}} \quad (4)$$

The normalised vector, z_{pca}, now follows a Gaussian distribution with zero mean and unit variance.

4.2. Autoencoder Accuracy

Encoder-Decoder Accuracy To evaluate the success of the PCA retargeting, we will first independently assess the encoder and decoder networks. To assess the encoder accuracy, all meshes in the test set will be passed through the encoder network and their latent space embeddings, $\hat{z}_{pca}$, retrieved. These will be compared to the ground-truth normalised shape parameters, z_{pca}, for the corresponding meshes. The decoder reconstruction accuracy for

a given mesh from the normalised shape parameters will be determined by passing these parameters through the decoder network. The reconstruction error will be calculated as the per-vertex euclidean distance between the ground truth and reconstructed meshes and compared to the reconstruction error for the PCA model.

End-to-end Reconstruction Accuracy To evaluate the model in a holistic manner, each mesh in the training set will be fed through the autoencoder model and the end-to-end reconstruction error will be calculated. This error will be compared to the reconstruction accuracy of the PCA model under the same circumstances; each mesh in the test set will be projected into the shape space of the PCA model and the retrieved shape parameters will be used to obtain the model reconstruction.

4.3. "Free" Latent Variables

Experiments will be conducted using 16 and 64 principal components to better understand how the number of PCA components impacts the presence of high-fidelity mesh details. "Free" variable lengths of 4, 8, and 16 will be assessed and their conformity to a Gaussian distribution will be evaluated. The reconstruction error for all meshes in the test set will be calculated and compared to that of the standard autoencoder model and the PCA model.

High Frequency Details The presence of high-fidelity mesh details will be evaluated via the Gaussian curvature of the mesh. The Gaussian curvature, $\mathcal{K}$, for a given vertex on a mesh surface is give as a product of the principal curvatures, κ_1 and κ_2, at that point Meyer et al. (2003). By calculating the Gaussian curvature at each point for all meshes reconstructed using the autoencoder, the autoencoder with "free" latent variables, and the PCA model, we can ascertain whether the introduction of the the "free" latent variables permits an increase in reconstruction detail and, if so, in which mesh regions.

5. Results

5.1. PCA Models

PCA models were constructed from a training dataset of $8,233$ densely registered facial samples. Four models were constructed which contained 16, 32, 64, and 128 principal shape components and retained 91.94%, 96.25%, 98.36%, and 99.30% of the overall shape variance, respectively. The test dataset contained 915 samples.

5.2. Model Architecture

Following a grid search, a learning rate of 0.001 with Adam optimisation was used. Optimal reconstruction and regularisation weights were determined to be 0.9 (λ_{rec}) and 1.2 (λ_{reg}), respectively. Filter sizes of $[16, 16, 32, 32, 32]$ were used in the encoder layer of all models. The reverse was applied to the decoder. All grid searches were performed using a mesh autoencoder model with 16 PCA and 0 "free" latent variables.

5.3. Autoencoder Accuracy

Encoder, decoder and end-to-end reconstruction accuracy for the retargeted autoencoders and PCA models are given in Table 5.3. As the PCA weights for all training and testing

instances were taken as ground truth for the model encoding, encoder accuracies are not reported for the PCA model. Model sizes are also recorded, where autoencoder models are shown to have far fewer parameters than the corresponding PCA models. For smaller latent vector sizes, the retargeted autoencoder performs almost identically PCA model in both decoder and end-to-end reconstruction accuracy. As the size of the latent vector increases, the PCA model slightly outperforms that retargeted autoencoder. This may be partially attributed to the higher encoder error when a larger latent vector size was used. Differences between decoder and end-to-end reconstruction accuracies are small. This can likely be attributed to the low encoder error at all latent vector sizes. The encoder error was shown to increase in an approximately linear fashion with the size of the latent vector.

Interpolation through the latent space is smooth and demonstrates good agreement between both PCA and AE models as shown in Figure 5.3. The greatest errors occurred at the shape extremes of the model principal components.

Model	LVs	Encoder	Decoder (mm)	E2E (mm)	Params.
PCA	16	N/A	0.72 ± 0.50	0.72 ± 0.50	$1,364K$
	32	N/A	0.49 ± 0.35	0.49 ± 0.35	$2,729K$
	64	N/A	0.32 ± 0.24	0.32 ± 0.24	$5,458K$
	128	N/A	0.21 ± 0.17	0.21 ± 0.17	$10,917K$
AE	16	0.017 ± 0.019	0.71 ± 0.49	0.71 ± 0.50	$90K$
	32	0.025 ± 0.031	0.49 ± 0.36	0.49 ± 0.36	$119K$
	64	0.044 ± 0.051	0.33 ± 0.25	0.33 ± 0.25	$176K$
	128	0.088 ± 0.102	0.24 ± 0.19	0.25 ± 0.20	$291K$

Table 1: Encoder, decoder, and end-to-end accuracy for the assessed latent variables sizes. The number of model parameters is also reported.

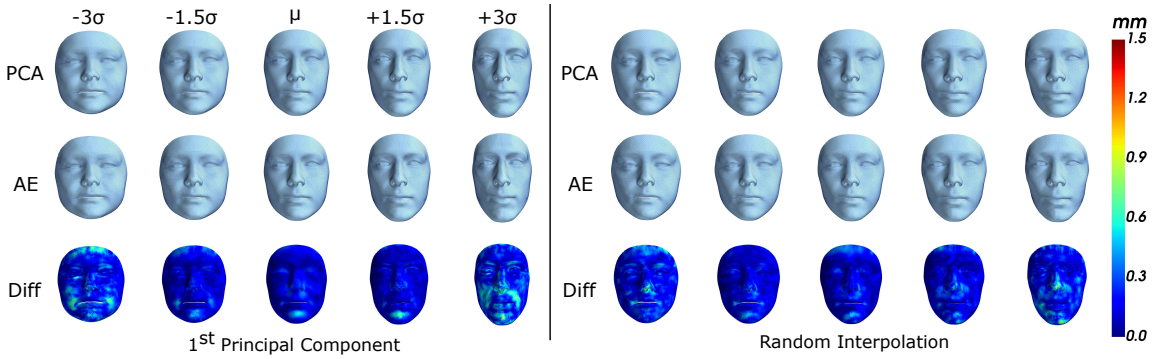

Figure 2: Interpolation in the latent space from -3σ to $+3\sigma$ along the first principal component (left) and between two shape instances randomly sampled from a multivariate Gaussian distribution (right).

5.4. "Free" Latent Variables

The impact of introducing the "free" latent variables is reported in Table 5.4. All "free" latent variables were set to zero when evaluating the decoder accuracy in isolation. The accuracy of the PCA latent variable encodings was unaffected by the addition of the "free" latent variables, as was the accuracy of the decoder in isolation. When 16 PCA latent variables were used, the addition of the "free" latent variables allowed for an increased end-to-end reconstruction accuracy. This accuracy increased for larger "free" latent variable sizes. When 64 PCA latent variables were used, the introduction of the "free" latent variables did not have any impact. In both cases, the "free" latent variables conformed to a Gaussian distribution with zero mean and unit variance.

To assess the impact of the "free" latent variables on the fidelity of mesh reconstructions, the Gaussian curvature of all reconstructed samples from the test set was calculated. As only the magnitude of the curvature, $\mathcal{K}$, was of interest, the absolute value was used when calculating curvature statistics. All autoencoders reported a higher mean Gaussian curvature than the PCA models. The Gaussian curvature for both PCA and autoencoder models was also higher when a larger number of latent variables were used. The similar curvature values for all autoencoder models indicates that the inclusion of the "free" latent variables did not lead to higher fidelity mesh reconstructions.

The shape changes captured by the "free" latent variables for a model with 16 PCA and 4 "free" latent variables are shown in Figure 5.4.

Model	LVs	Encoder	Decoder (mm)	E2E (mm)	Gauss. Curv.
PCA	16	N/A	0.72 ± 0.50	0.72 ± 0.50	0.0679 ± 0.4681
	64	N/A	0.32 ± 0.24	0.32 ± 0.24	0.0681 ± 0.4681
AE	16	0.017 ± 0.019	0.71 ± 0.49	0.71 ± 0.50	0.0704 ± 0.4680
0 free LVs	64	0.044 ± 0.051	0.33 ± 0.25	0.33 ± 0.25	0.0707 ± 0.4679
AE	16	0.017 ± 0.018	0.72 ± 0.50	0.65 ± 0.46	0.0707 ± 0.4679
4 free LVs	64	0.044 ± 0.052	0.33 ± 0.25	0.33 ± 0.25	0.0708 ± 0.4678
AE	16	0.017 ± 0.020	0.72 ± 0.50	0.61 ± 0.43	0.0705 ± 0.4680
8 free LVs	64	0.043 ± 0.051	0.33 ± 0.25	0.33 ± 0.25	0.0708 ± 0.4679
AE	16	0.016 ± 0.019	0.72 ± 0.50	0.54 ± 0.39	0.0705 ± 0.4680
16 free LVs	64	0.043 ± 0.050	0.33 ± 0.25	0.33 ± 0.25	0.0707 ± 0.4679

Table 2: Encoder, decoder and end-to-end (E2E) model accuracy for the assessed latent variables sizes. Gaussian Curvature is reported for E2E reconstructions.

6. Findings

The primary hypothesis of this work was that a linear 3DMM constructed using principal component analysis could be retargeted as a non-linear convolutional mesh autoencoder. As the reconstruction errors for the test dataset were very similar for the PCA and autoencoder models, we conclude that the results support this hypothesis. Decoder and end-to-end autoencoder accuracy were also noted to be very similar to each other. This result, combined with the low encoder error across all latent vector sizes, indicate a high degree of accuracy in

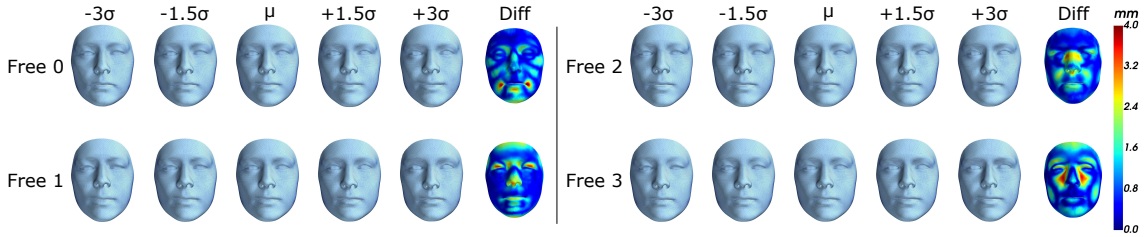

Figure 3: Shape changes captured by the "free" latent variables from -3σ to $+3\sigma$. Heatmaps show the per-vertex Euclidean distance between these extremes.

the encoding of the PCA latent vector into the latent space of the autoencoder. While the decoder error decreases as more components are added to the autoencoder latent space, the encoder error increases when larger latent vectors sizes are used. One reason for this may be that as the latent vector size increases, it becomes more difficult to maintain high levels of accuracy across the larger number latent variables. A second reason is that successive components in the latent vector represent increasingly small directions of shape variance. With such minimal shape changes it may be more difficult for the model to accurately learn these shape variations and how they are related to the encoder values.

The second hypothesis tested in this work was that the introduction of free latent variables in the latent layer of the convolutional mesh autoencoder would facilitate more accurate mesh reconstructions than could be achieved using PCA latent variables alone. This was shown to be the case when lower PCA latent vector sizes were used, but the impact of the "free" latent variables was negligible for larger PCA latent vector sizes were used. This is explained by the percentage of shape variance has not been captured by the PCA components at these latent vector sizes. The model with 16 PCA components expresses only 91.94% of the total shape variance of the training set, and the "free" latent variables are more effective at increasing the model capacity and reconstruction accuracy. When 64 components are used, however, these already express 98.36% of the shape variance and there is less scope for the "free" variables to improve the reconstruction quality. Gaussian curvature experiments demonstrated that the inclusion of the "free" latent variables had little impact on the fidelity of the mesh reconstructions. All autoencoders were shown to produce higher fidelity meshes than their PCA counterparts using this metric.

In conclusion, this work demonstrates the successful conversion of a linear PCA model to a non-linear mesh autoencoder, allowing for equivalent reconstruction accuracy at a smaller model size. The "free" latent variables allows for more accurate mesh reconstructions at smaller PCA latent vector sizes, but do not impact the fidelity of the reconstructed meshes.

7. Documented Modifications

An additional grid search was performed to determine the optimal weight for the "free" latent variable component of the loss term, λ_{kl}. A model with 16 PCA and 8 "free" latent variables was used. This was required to prevent posterior collapse during training, where the decoder ignored the values of z_{free} in the latent vector, resulting in a model that

performed equivalently to one trained using PCA latent variables only. The optimal KL weight term, λ_{kl}, was determined to be 0.001. The updated loss term took the form:

$$loss = \lambda_{rec}||x - \hat{x}|| + \lambda_{reg}(||z_{pca} - \hat{z}_{pca}|| + \lambda_{kl} KL[\mathcal{N}(\mu_{z_{free}}, \sigma_{z_{free}}), \mathcal{N}(0, 1)]) \quad (5)$$

References

Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *Proceedings of the 26th annual conference on computer graphics and interactive techniques*, SIGGRAPH '99, pages 187–194, Jul 1, 1999.

James Booth, Anastasios Roussos, Allan Ponniah, David Dunaway, and Stefanos Zafeiriou. Large scale 3D morphable models. *International Journal of Computer Vision*, 126(2):233–254, Apr 2018.

Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael M. Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. In *Neural Information Processing Systems*, 2016.

Giorgos Bouritsas, Sergiy Bokhnyak, Stylianos Ploumpis, Stefanos Zafeiriou, and Michael Bronstein. Neural 3D morphable models: Spiral convolutional networks for 3D shape representation learning and generation. In *International Conference on Computer Vision (ICCV)*, pages 7212–7221. IEEE, 2019.

Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4): 18–42, 2017.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014)*, 2014.

Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. Facewarehouse: A 3D facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, 2014.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *International Conference on Neural Information Processing Systems*, page 2180–2188, 2016.

Shiyang Cheng, Irene Kotsia, Maja Pantic, and Stefanos Zafeiriou. 4dfab: A large scale 4D facial expression database for biometric applications. *Conference on Computer Vision and Pattern Recognition*, pages 5117–5126, 2018.

Hang Dai, Nick Pears, William Smith, and Christian Duncan. A 3D morphable model of craniofacial shape and texture variation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3104–3112. IEEE, 2017.

T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, J. M. Tomczak, A. Globerson, and R. Silva. Hyperspherical variational auto-encoders. In *Uncertainty in Artificial Intelligence*, pages 856–865. AUAI Press, 2018.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Conference on Neural Information Processing Systems (NIPS)*, 2016.

Frederick Diehl, Thomas Brunner, Michael Truong Le, and Alois Knoll. Towards graph pooling by edge contraction. In *ICML 2019 Workshop on Learning and Reasoning with Graph-Structured Data*, 2019.

Matthias Fey, Jan E. Lenssen, Frank Weichert, and Heinrich Muller. SplineCNN: Fast geometric deep learning with continuous B-Spline kernels. *Conference on Computer Vision and Pattern Recognition*, pages 869–877, 2018.

Benoit Gaujac, Ilya Feige, and David Barber. Learning disentangled representations with the wasserstein autoencoders, Oct 7, 2020. URL https://arxiv.org/abs/2010.03459.

Shunwang Gong, Lei Chen, Michael Bronstein, and Stefanos Zafeiriou. SpiralNet++: A fast and highly efficient mesh convolution operator. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, 2019.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinivk, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/pdf?id=Sy2fzU9gl.

Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. *Transforming Auto-Encoders*, volume 6791 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 9783642217340.

R. D. Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1808.06670.

Weonyoung Joo, Wonsung Lee, Sungrae Park, and Il-Chul Moon. Dirichlet variational autoencoder. *Pattern recognition*, 107:107514, Nov 2020. URL http://dx.doi.org/10.1016/j.patcog.2020.107514.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations (ICLR)*, Sep 9, 2016.

Tejas D Kulkarni, William F. Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, volume 28, pages 2539–2547, 2015.

Ron Levie, Federico Monti, Xavier Bresson, and Michael M. Bronstein. CayleyNets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing*, 67(1):97–109, Jan 1, 2019.

Isaak Lim, Alexander Dielen, Marcel Campen, and Leif Kobbelt. A simple approach to intrinsic correspondence learning on unstructured 3D meshes. In *Proceedings of the European Conferenceon Computer Vision Workshops (ECCVW)*, 2018.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael Black. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics (TOG)*, 34(6):1–16, 2015.

Marcel Lüthi, Christoph Jud, Thomas Gerig, and Thomas Vetter. Gaussian process morphable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(8):1860–1873, 2018.

Jonathan Masci, Davide Boscaini, Michael M. Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW*, pages 832–840. IEEE, 2015.

Emile Mathieu, Tom Rainforth, N. Siddharth, and Yee Whyte Teh. Disentangling disentanglement in variational autoencoders. *Proceedings of Machine Learning Research*, 97, Jun 9, 2019.

Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. *Discrete Differential-Geometry Operators for Triangulated 2-Manifolds*. Springer Berlin Heidelberg, 2003. URL https://search.datacite.org/works/10.1007/978-3-662-05105-4_2.

Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M. Bronstein. Fake news detection on social media using geometric deep learning, Feb 10, 2019. URL https://arxiv.org/abs/1902.06673.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

Stylianos Ploumpis, Evangelos Ververas, Eimear O' Sullivan, Stylianos Moschoglou, Haoyang Wang, Nick Pears, William Smith, Baris Gecer, and Stefanos P. Zafeiriou. Towards a complete 3D morphable model of the human head. *IEEE transactions on pattern analysis and machine intelligence*, PP:1, Apr 29, 2020.

Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D faces using convolutional mesh autoencoders. In *European Conference on Computer Vision*, 2018. URL http://arxiv.org/abs/1807.10267.

Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, page II–1431–II–1439, 2014.

Danilo J. Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*, 2014.

Javier Romero, Dimitrios Tzionas, and Michael Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics (TOG)*, 36(6):1–17, 2017.

Huajie Shao, Haohong Lin, Qinmin Yang, Shuochao Yao, Han Zhao, and Tarek Abdelzaher. DynamicVAE: Decoupling reconstruction error and disentangled representation learning, Sep 14, 2020. URL https://arxiv.org/abs/2009.06795.

Qiaoyu Tan, Ninghao Liu, and Xia Hu. Deep representation learning for social network analysis. *Frontiers in Big Data*, 2, Apr 3, 2019.

Shanshan Tang, Bo Li, and Haijun Yu. Chebnet: Efficient and stable constructions of deep neural networks with rectified power units using chebyshev approximations, Nov 7, 2019. URL https://arxiv.org/abs/1911.05467.

Michael Tschannen, Josip Djolonga, Paul K. Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1907.13625.

Kirill Veselkov, Guadalupe Gonzalez, Shahad Aljifri, Dieter Galea, Reza Mirnezami, Jozef Youssef, Michael Bronstein, and Ivan Laponogov. Hyperfoods: Machine intelligent mapping of cancer-beating molecules in foods. *Scientific reports*, 9(1):9237–12, 2019.

Jiacheng Xu and Greg Durrett. Spherical latent spaces for stable variational autoencoders. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018. URL https://arxiv.org/abs/1808.10805.

Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *International Conference on Neural Information Processing*, pages 4805–4815, 2018.