Learning Feature Aggregation for Deep 3D Morphable Models - Supplementary Material

Zhixiang Chen
Imperial College London
zhixiang.chen@imperial.ac.uk

Tae-Kyun Kim
Imperial College London and KAIST
tk.kim@imperial.ac.uk

Table 1. Architecture of the encoder

layer	input size	output size
convolution	n_4*3	n_4*16
downsampling	n_4*16	n_3*16
convolution	n_3*16	n_3*16
downsampling	n_3*16	n_2*16
convolution	n_2*16	n_2*16
downsampling	n_2*16	n_1*16
convolution	n_1*16	n_1*32
downsampling	n_1*32	n_0*32
fully connected	n_0*32	n_z

Table 2. Architecture of the decoder

layer	input size	output size
fully connected	n_z	n_0*32
upsampling	n_0*32	n_1*32
convolution	n_1*32	n_1*32
upsampling	n_1*32	n_2*32
convolution	n_2*32	n_2*16
upsampling	n_2*16	n_3*16
convolution	n_3*16	n_3*16
upsampling	n_3*16	n_4*16
convolution	n_4*16	n_4*16
convolution	n_4*16	n_4*3

In this supplementary material, we present more details about the experiments in the main paper and show additional experimental results to evaluate our proposed feature aggregation method. In Sec. 1, we provide details of our evaluated architecture. Sec. 2 provides the statistical information of the datasets. Sec. 3 contains implementation details to train the models. Sec. 4 presents more results about the experiments in the main paper. In Sec. 5, we give the experimental results on different network architectures. In Sec. 6, we show further ablation study experiments on the attention mechanism. In Sec. 7, we provide experimental results of parameter sensitivity studies. We give the model complexity analysis in Sec. 8.

1. Architecture details

Our model consists of an encoder and a decoder. The architecture details of the encoder and decoder are listed in Tables 1 and 2, resepectively. The convolution is Chebyshev convolution filter with $K = 6$ Chebyshev polynomials for CoMA and spiral convolution of 1 hop for Neural3DMM. The aggregation, including downsampling and upsampling, is either implemented by QEM as in [6] or accomplished by our proposed attention based module. The dimension of latent representation n_z is set as one of $\{8, 16, 32, 64\}$ in the evaluated settings. The numbers of vertices at hierarchical levels are summarized in Table 3.

Table 3. Number of vertices at hierarchical levels

dataset	n_0	n_1	n_2	n_3	n_4
COMA [6]	20	79	314	1256	5023
DFAUST [1]	27	108	431	1723	6890
SYNHAND [4]	5	19	75	299	1193

Table 4. Summary of statistics of the benchmark datasets

name	#mesh	#vertex	#ID	#pose/expression
COMA	20K	5023	12	12
DFAUST	40K	6890	10	14
SYNHAND	100K	1193	-	-

2. Datset statistics

Table 4 summarizes the statistics of the datasets used for evaluating 3D models. Since the deformations are randomly generated, there is no identity and pose category information for the SYNHAND dataset.

3. Implementation details

We implement the models with PyTorch [5]. We adopt the training settings suggested by the original authors [6, 2]. We train the CoMA and Deep3DMM (spectral) models for 300 epochs with learning rate of 8e-3. We train the Neural3DMM and Deep3DMM (spiral) models for 200 epochs

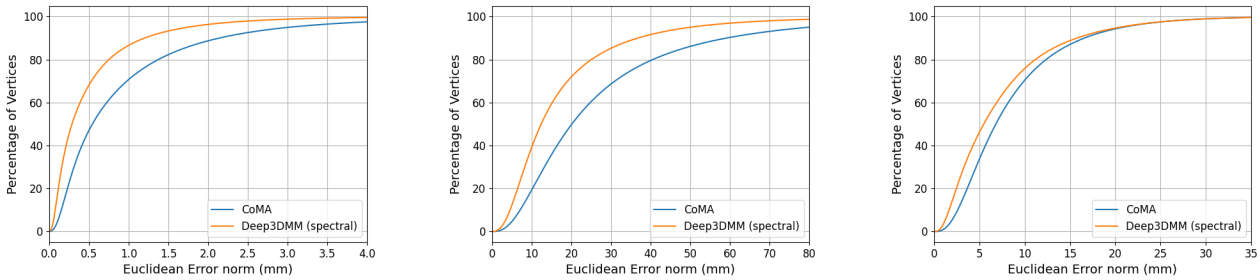

Figure 1. Cumulative Euclidean errors of CoMA method and our Deep3DMM (spectral) method with proposed feature aggregation module on COMA(left), DFAUST(middle), and SYNHAND(right) datasets

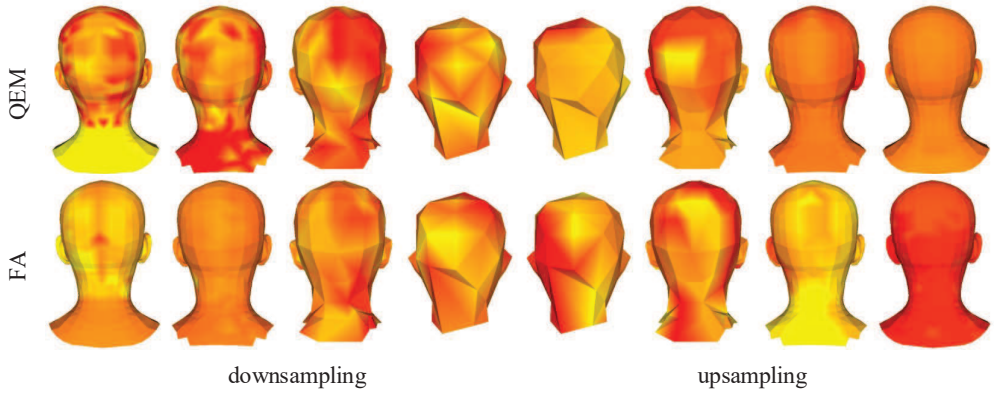

Figure 2. Back view of the visualization of mapping matrices of down-sampling and up-sampling on COMA dataset with t-SNE (best viewed in color). The first and second rows are the results of quadric error minimization in [6, 2] and our proposed feature aggregation, respectively

with learning rate of 1e-3.

4. More experimental results

In this section, we provide more experimental results of the reconstruction errors and show more visualizations of the mapping matrices.

4.1. Cumulative Euclidean errors

In Fig. 1, we show the cumulative distribution of the Euclidean errors with and without our feature aggregation module for CoMA model with latent dimension of 8. We can find that for a given error threshold, more vertices can satisfy the constraint with lower error by applying our feature aggregation module. This is consistent with the observation from the qualitative results in the main paper.

4.2. More Visualization of the mapping matrices

In Fig. 2, we show the back view of the mapping matrices on COMA dataset, which is complementary to the front view in the main paper. The pattern is similar to the front view in the main paper. In Figs. 3 and 4, we directly show the values of the mapping matrices on COMA dataset. Since there are large number of columns and rows in each mapping matrix, a better view can be obtained by zooming in on the figures. While the mapping matrices obtained by QEM and our feature aggregation mechanism demonstrate similar pattern in positions of the dominant elements, the values of these elements are different for these two methods. This is because our proposed feature aggregation mechanism enables learning the weights from the training data automatically. Moreover, the mapping matrices learned by FA are more dense than those computed by QEM. This shows that our proposed feature aggregation mechanism also learns the receptive fields.

4.3. Qualitative results with spiral convolutions

In Fig. 5, we show the per vertex Euclidean error of different morphable models on several shapes from the three datasets for qualitative comparison. The latent dimension is set as 8. We can find that our model can reduce the large errors of the compare model (red regions) by providing accurate predictions. Our model can also recover more details than the compared model, leading to more realistic shapes

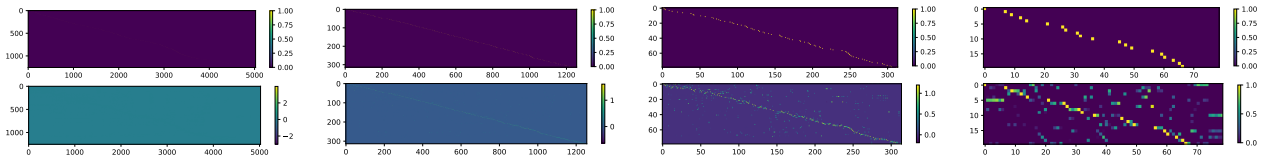

Figure 3. Visualization of mapping matrices of downsampling (top row) and upsampling (bottom row) with existing QEM method on COMA dataset(best viewed in color and zoom in to see details)

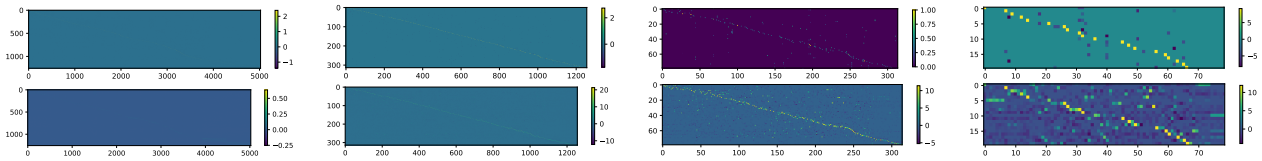

Figure 4. Visualization of mapping matrices of downsampling (top row) and upsampling (bottom row) with our proposed feature aggregation module on COMA dataset(best viewed in color and zoom in to see details)

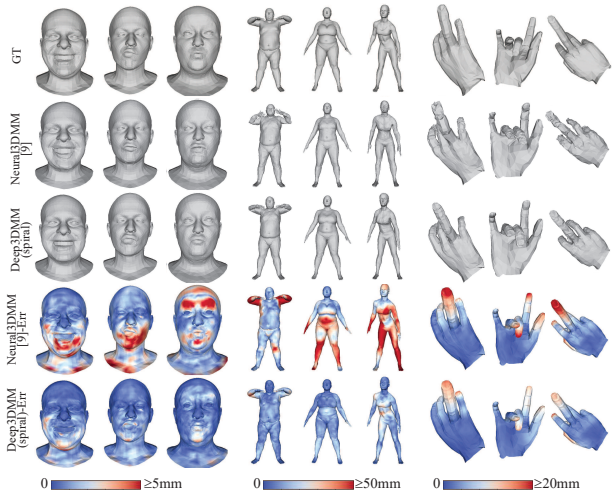

Figure 5. Qualitative comparison of spiral convolution based models on COMA (left), DFAUST (middle), and SYNHAND (right) datasets. The first row is the ground truth shapes. The second and third rows show the reconstructed shapes, while the fourth and fifth rows show the corresponding reconstruction errors

and lower reconstruction errors over almost all regions.

4.4. Extrapolation experiment

To further measure the generalization capability of our model, we follow the setting in [6] to reconstruct expressions that are excluded from the training set. We conduct 12 different experiments to evaluate the performance on each of the 12 expressions when the remaining 11 expressions are used as the training samples. We compare the results (mean, standard deviation and median of the Euclidean distance) of our model with CoMA [6], PCA and FLAME [3], as shown in Table 5. The results of the compared methods are taken from [6]. We can observe that our model achieve better performance than the compared methods on all expression sequences.

5. Results with different network architectures

In this section, we evaluate our proposed feature aggregation module on different variants of network architecture, including the number of convolution filters of the spiral convolution and the Chebyshev polynomial order of the spectral convolution.

5.1. Number of convolution filters

To explore the effectiveness of our proposed feature aggregation module with different network architectures, we conduct experiments on two settings of the number of convolution filters. The simple setting denotes the network architecture introduced in Tables 1 and 2, where the number of filters are (3,16,16,16,32) and (32,32,16,16,16,3) for the encoder and decoder, respectively. The wider setting denotes a larger number of filters, where they are (3,16,32,64,128) and (128,64,32,32,16,3) for the encoder and decoder, respectively. Table 6 shows the reconstruction errors on COMA dataset with the latent dimension of 8 for Neural3DMM and out spiral convolution based Deep3DMM. We can see that our model performs better than the baseline model in both scenarios. Note that our model has the same inference parameters as the compared model. Our model only introduces $8 + (5023 + 1256 + 1256 + 314 + 314 + 79 + 79 + 20) \times c$ parameters for the keys and querys at the training stage.

5.2. Chebyshev polynomial order

In Table 7, we show the results for variant Chebshev polynomial order K. The experiments are again conducted on COMA dataset with the latent dimension as 8. We can see that model with our feature aggregation module can consistently perform better.

Table 5. Reconstruction errors on extrapolation setting

	Deep3DMM(spectral)		CoMA [6]		PCA		FLAME [3]	
Sequence	Mean Error	Median	Mean Error	Median	Mean Error	Median	Mean Error	Median
bareteeth	**1.190±1.524**	**0.691**	1.376±1.536	0.856	1.957±1.888	1.335	2.002±1.456	1.606
cheeks in	**1.071±1.322**	**0.646**	1.288±1.501	0.794	1.854±1.906	1.179	2.011±1.468	1.609
eyebrow	**0.851±1.011**	**0.505**	1.053±1.088	0.706	1.609±1.535	1.090	1.862±1.342	1.516
high smile	**1.037±1.164**	**0.614**	1.205±1.252	0.772	1.841±1.831	1.246	1.960±1.370	1.625
lips back	**1.060**±1.590	**0.580**	1.193±**1.476**	0.708	1.842±1.947	1.198	2.047±1.485	1.639
lips up	**0.902±1.114**	**0.497**	1.081±1.192	0.656	1.788±1.764	1.216	1.983±1.427	1.616
mouth down	**0.847±1.062**	**0.517**	1.050±1.183	0.654	1.618±1.594	1.105	2.029±1.454	1.651
mouth extreme	**1.139±1.468**	**0.640**	1.336±1.820	0.738	2.011±2.405	1.224	2.028±1.464	1.613
mouth middle	**0.745±0.934**	**0.439**	1.017±1.192	0.610	1.697±1.715	1.133	2.043±1.496	1.620
mouth open	**0.741±0.996**	**0.431**	0.961±1.127	0.583	1.612±1.728	1.060	1.894±1.433	1.544
mouth side	**1.103**±1.711	**0.567**	1.264±**1.611**	0.730	1.894±2.274	1.132	2.090±1.510	1.659
mouth up	**0.835±0.983**	**0.501**	1.097±1.212	0.683	1.710±1.680	1.159	2.067±1.485	1.680

Table 6. Reconstruction errors with different settings of convolution filters

method	simple	wider
Neural3DMM	0.785	0.525
Deep3DMM (spiral)	0.487	0.420

Table 7. Reconstruction errors with different order K of Chebyshev polynomial

method	CoMA	Deep3DMM (spectral)
K=6	0.939	0.519
K=3	1.031	0.558

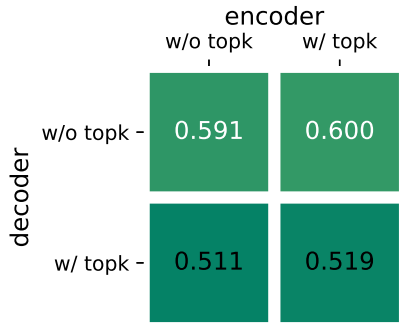

Figure 6. Reconstruction errors w/ and w/o applying the topk selection in FA module

6. Ablation studies

In this section, we provide ablation studies to show the effect of each component in the model. The experiments are conducted on COMA dataset by using the spectral convolution with the latent dimension as 8.

6.1. Effect of the topk selection

In Fig. 6, we show the reconstruction errors with and without the topk selection for the mask operation in the feature aggregation module. As we can see, the error is reduced by applying the topk selection strategy in the decoder by a large margin. And the performance is slightly deteriorated by applying the topk selection strategy in the encoder. In this work, we choose to apply the topk selection on both the encoder and decoder for the consideration of the speed. By adopting the topk selection strategy, the generated mapping matrices are guaranteed to be sparse, which can be leveraged to accelerate both the training and the inference of the model.

Table 8. Reconstruction errors with different mapping matrices

method	QEM	No fusing	Fusing
Error	0.939	0.693	0.519

6.2. Effect of fusing mapping matrices

We also study the effect of fusing learned mapping matrices with precomputed mapping matrices. The results are shown in Table 8. By using the learned mapping matrices only, we can also significantly reduce the reconstruction error. Combining both mapping matrices by a linear fusion, we can further lower the reconstruction error. This is possible due to that the precomputed mapping matrices can benefit the training of the other components, including the convolution and the fully connected layers, especially at the early stage of the training phase.

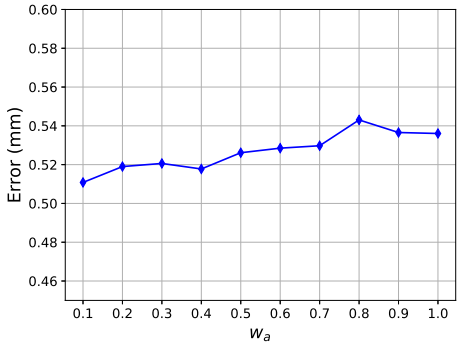

Figure 7. Reconstruction errors with different initialization value for the weight w_a

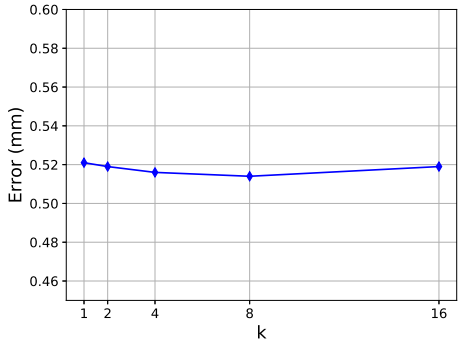

Figure 9. Reconstruction errors with different k for topk selection in the encoder

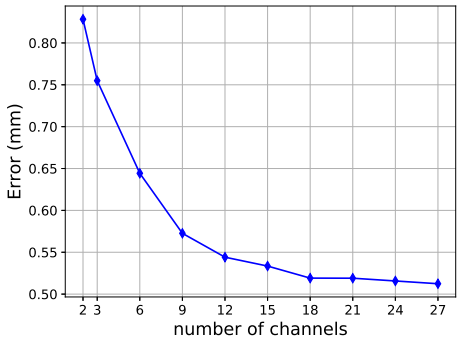

Figure 8. Reconstruction errors with different number of channels for the keys and queries in the feature aggregation module

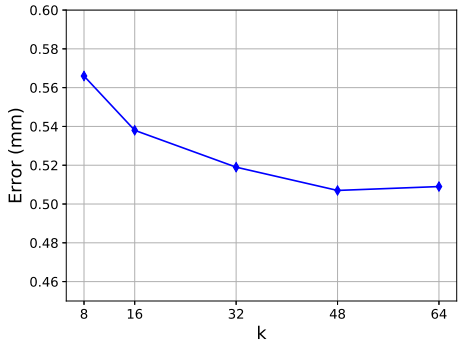

Figure 10. Reconstruction errors with different k for topk selection in the decoder

7. Parameter sensitivity studies

In this section, we provide parameter sensitivity studies to get better understanding of the proposed feature aggregation module. The experiments are conducted on COMA dataset by using the spectral convolution with the latent dimension as 8.

7.1. Initialization of the weight w_a

In Fig. 7, we show the reconstruction errors by training the model with different initialization values for the weight w_a. While a smaller weight can lead to better performance, the performance variation is not notable.

7.2. Number of channel c

In Fig. 8, we show the variation of the reconstruction error with respect to the dimension of channels of the keys and queries in the feature aggregation module. By increasing c from 2 to 18, we can observe a significant drop in the corresponding error. The performance is almost saturated when c is larger than 18. Note that we can surpass the QEM method even setting $c = 2$ for our feature aggregation module.

7.3. Top k for the encoder and decoder

In Figs. 9 and 10, we show the variation of the reconstruction error with respect to the k value in the encoder and decoder, respectively. By changing the k value in the encoder, the performance is only slightly influenced. In contrast, larger k in the decoder would lead to better generalization with lower error.

7.4. Initialization of the keys and queries

We also study the effect of different initialization schemes of the keys and queries. Table 9 gives the reconstruction errors of three different initialization, namely normal, uniform and template. In the normal and uniform settings, we initialize the keys and queries by the random normal and uniform distributions, respectively. In the precomputed setting, we use the vertex positions at each level computed by the mesh decimation to initialize the keys and queries. This is the initialization we adopt in this paper. The precomputed based initialization outperforms the others significantly.

Table 9. Reconstruction errors with different initializations for the keys and queries

method	normal	uniform	precomputed
error (mm)	0.649	0.619	0.519

8. Complexity

By directly parameterizing the mapping matrices, the complexity is with quadric scale $\mathcal{O}(n_l n_{l-1})$. By using our attention based mechanism, the complexity to model the mapping matrices is reduced to a linear scale $\mathcal{O}(n_l + n_{l-1})$ which parameterizes the keys and queries. Thus, we provide a feasible solution to circumvent the over-parameterziation problem.

References

[1] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: registering human bodies in motion. In *CVPR*, pages 5573–5582, 2017. 1

[2] Giorgos Bouritsas, Sergiy Bokhnyak, Stylianos Ploumpis, Stefanos Zafeiriou, and Michael M. Bronstein. Neural 3D morphable models: Spiral convolutional networks for 3D shape representation learning and generation. In *ICCV*, pages 7212–7221, 2019. 1, 2

[3] Tianye Li, Timo Bolkart, Michael J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *TOG*, 36(6):194:1–194:17, 2017. 3, 4

[4] Jameel Malik, Ahmed Elhayek, Fabrizio Nunnari, Kiran Varanasi, Kiarash Tamaddon, Alexis Héloir, and Didier Stricker. Deephps: End-to-end estimation of 3D hand pose and shape by learning from synthetic depth. In *International Conference on 3D Vision, 3DV*, pages 110–119, 2018. 1

[5] 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. *NeurIPS*, 2017. 1

[6] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D faces using convolutional mesh autoencoders. In *ECCV*, pages 725–741, 2018. 1, 2, 3, 4