A Deep Variational Autoencoder Approach for Robust Facial Symmetrization

Ting Wang[1]
wangting@sdust.edu.cn
Shu Zhang[2, *]
zhangshu@ouc.edu.cn
Junyu Dong[2]
dongjunyu@ouc.edu.cn
Yongquan Liang[1]
lyq@sdust.edu.cn

[1] Shandong University of Science and Technology
Qingdao, China
[2] Ocean University of China
Qingdao, China

Abstract

Face symmetrization has extended applications in both academic and medical fields. Human face possesses an important characteristic, which is as known as symmetry. However, in practice, this symmetry is never perfect, which yields a large amount of studies around this topic. For example, facial paralysis evaluation based on facial asymmetry analysis, facial beauty evaluation based on facial symmetry analysis, facial recognition, and facial frontalisation among others. Currently, there are still very limited researches that are dedicated for this topic. Most of the existing studies only utilized their own implantations for symmetric face generating to achieve their researches in other fields. Thus, limitations can be noticed in their methods, such as manual intervention requirement. Furthermore, most existing method utilize facial landmark detection algorithms for symmetric face construction. Despite the promising accuracy of the landmark detection algorithms, the uncontrolled conditions in facial images can badly impact the performance of the symmetric face production. To this end, this paper presents a variational autoencoder based deep generative model for symmetric face generating. It is achieved by a 3-stage training process to avoid the demand for the large size of the symmetric face as training data. Experiments are conducted with comparisons with several methods that achieved by some of the most popular facial landmark detection algorithms. Competitive results are achieved.

Key words, facial symmetry, multi-stage training, variational autoencoder.

1 Introduction

Human face has an important character, which is as known as symmetry. However, in practice, this facial symmetry is never perfect. Difference can always be spot between left and right half faces, which yields a number of studies around this topic. For example, facial paralysis evaluation based on facial asymmetry analysis, facial beauty evaluation based on facial symmetry analysis, and many others. To properly evaluate this symmetry of the face, a facial symmetrization process is commonly utilized in most of the existing researches,

[*] Shu Zhang is the corresponding author.

which provides a perfectly symmetric face based on only one half of the face. Therefore, the performance of the symmetrization is critical to those facial symmetry based applications.

Most existing methods for face symmetrization are based on the facial geometric calculations with extracted facial landmark locations. However, the accurate of those extracted landmarks can drop dramatically under certain circumstances. To avoid such error accumulation during the process, this paper presents a novel method for a perfectly symmetric face generating by whole facial image analysis. A deep generative neural network is proposed for this task based on a deep Variational Autoencoder (VAE) architecture. One of the advantage of the proposed method is that it can minimize the influence of the unwanted conditions in the facial image when symmetrize the face, such as the facial illuminations, the occlusions of the face, and many others. Those unwanted conditions can badly damage the performance of the facial landmark localization, hence affect the facial symmetrization process for traditional methods.

The contributions of this paper are as follows: (a) A generative model based on the deep variational autoencoder network is constructed for facial image synthesis, which consists of a CNN (Convolutional Neural Network) as encoder and a DGDN (Deep Generative Deconvolutional Network) as decoder. (b) A multi-stage training process that can learn the model parameters in a coarse-to-fine manner, which avoids the demand for large amount of the symmetric faces as training data. (c) A competitive performance demonstrated in the experiment compared with other existing methods.

2 Related work

The generating of the symmetric face has a great number of applications in many fields related to the computer vision. For example, most computer vision based facial palsy studies are deeply coupled with facial symmetry analysis to visually diagnose the facial regional movement functions [1–6]. Apart from this, the facial symmetry is also an important factor to evaluate the facial beauty or facial attractiveness reported in many studies [7–10]. The analysis of the facial symmetry can also contribute to many computer vision based applications, such as facial frontalisation [11–15], facial feature detection [16], facial recognition [17] and many other facial manipulations [18–21].

Despite such large academic demands, there are still very limited researches dedicated for symmetric face generating. Most of the studies only achieves their own implementations of symmetric face generating for their academic studies in other fields. For example, Hassner *et al.* [12] have explored the face frontalisation problem with the consideration of facial symmetry. Their facial symmetry model, called soft-symmetry, was achieved with the facial features extracted by Supervised Descent Model (SDM) [22], which is one of the most popular methods for facial landmark localization. Iacopo *et al.* [23] have presented their work to utilize the facial symmetry property for pose-aware face recognition. They employed the similar strategy as Hassner *et al.* did in their frontalisation study, which utilized SDM facial landmark detection algorithm. Xu *et al.* [17] have also discussed the facial recognition based on the facial symmetry property. Their method was based on the generated virtual facial images that were axis-symmetric. Those virtual facial images were synthesized by flipping the original face that were manually cantered in the images. Their method required manual interventions during the symmetric face preparation step. Harguess *et al.* [24] have proposed a study to explore the relationship between the facial symmetry and face recognition. They included the discussions to utilize an average-half-face [25] to obtain a symmetric face for the study. To achieve the average-half-face, the facial image was

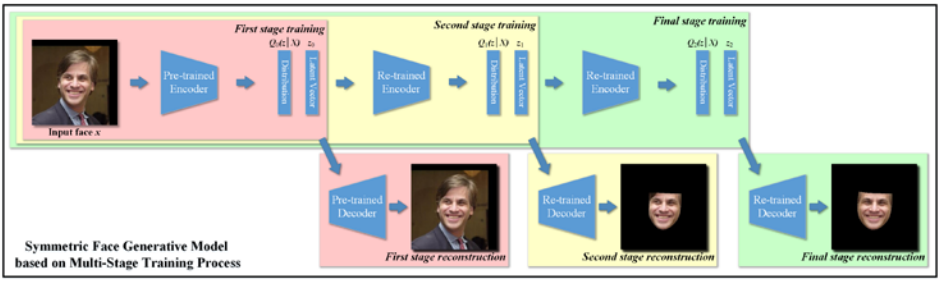

Figure 1: The pipeline of the proposed model.

firstly centred about the nose, then divided into two symmetric halves, and finally averaged altogether. However, in their paper, they didn't mention about how to localize the nose automatically. In practice, this can be done by either manually annotations or facial landmark detection algorithms, such as AAM [26], SDM [22], CLM [27] or CLNF [28]. Passalis *et al*. [29] have presented their research using facial symmetry to handle pose variations for face recognition. They utilized a facial landmark model that extracted eight anatomical facial landmarks for facial symmetry model generating.

To summarize up, almost all the existing researches that related to symmetric face generating utilize a certain facial landmark extraction algorithm to find the facial symmetric axis for symmetric face reconstruction. Despite that the performance of the facial landmark extraction algorithms is increasing over recent decades, the facial landmark based facial symmetric axis estimation can still be less accurate since there are so many uncontrolled conditions in the facial images, such as head poses and occlusions. There is virtually no method that extract symmetric face based on full image analysis. Those challenges can provide an opportunity and a motivation in the computer vision community for the researchers to conduct more comprehensive studies on this topic.

3 Symmetric face generative model composition

As introduced in [30], the variational autoencoder (VAE) network commonly consists of two parts: an encoder network and a decoder network. To recall the network structure in a simple way, the encoder compresses the input data x (an image) into a vector z of latent variables, which represents the most important features from x. The decoder translates this latent vector z back to a desired image $\hat{x}$, more specifically, according to the sampled z based on a target distribution $Q(z|X)$ learnt from the training data. The proposed method is inspired by the concept of VAEs. The information of the symmetric face is obviously encoded in the original facial image, which can be extracted from z, a vector of the compressed input data. It doesn't necessarily be the whole latent vector, but a sample of it. The goal of the proposed method is to estimate this sampling distribution, which can construct a model for face symmetrization.

The main work flow of the proposed method is demonstrated in Fig. 1, which consists of three training stages: (**a**) The first stage tries to reconstruct the original facial image, which can be trained using the face training data that are sufficient enough in publicly available database. This stage will help to extract a proper z to accurately describe the input data. (**b**) The second stage tries to re-train the model to find the facial regions in the facial images. This stage will learn a distribution to accurately sample z for image features that represent the faces. (**c**) The third stage tries to transfer the model to achieve the symmetric face generating. The training data in the second and third stages can be substantially less than what in the first stage. The proposed model is achieved by a convolutional neural network as the encoder, and a deep generative deconvolutional network as the decoder.

3.1 CNN based face encoder

The encoder of the proposed model is constructed using a convolutional neural network. Considering N data points (facial images) $X = \{x^{(i)}\}_{i=1}^{N}$, we are trying to estimate a probability $Q(z|X)$ to properly describe the distribution of the latent vector z given the input data points. This is achieved by a convolutional layer and a pooling layer alternating multiple times. An activation operation is applied at the end of each convolutional layer. The parameters of each convolutional layer and pooling layer can be different and make the stack of the layers forming up a top-down funnel shape. The convolutional and pooling layers are illustrated below with the total layer number as L and current layer as layer-l:

$$\text{Conv Layer:} \quad \tilde{\mathbf{C}}^{(i,k_{l-1},l-1)} = \mathbf{C}^{(i,l-2)} * \mathbf{F}^{(k_{l-1},l-1)}, k_{l-1} \in \{1, \dots, K_{l-1}\} \tag{1}$$

$$\text{Pooling Layer:} \quad \mathbf{C}^{(i,l-1)} \sim \text{pooling}[h\left(\tilde{\mathbf{C}}^{(i,\Sigma K_{l-1},l-1)}\right)] \tag{2}$$

$$\text{Conv Layer:} \quad \tilde{\mathbf{C}}^{(i,k_l,l)} = \mathbf{C}^{(i,l-1)} * \mathbf{F}^{(k_l,l)}, k_l \in \{1, \dots, K_l\} \tag{3}$$

$$\text{Pooling Layer:} \quad \mathbf{C}^{(i,l)} \sim \text{pooling}[h\left(\tilde{\mathbf{C}}^{(i,\Sigma K_l,l)}\right)] \tag{4}$$

$$\text{Latent Vector:} \quad z^{(i)} \sim Q(\mu_\phi\left(\mathbf{C}^{(i,L)}\right), \log \sigma_\phi^2\left(\mathbf{C}^{(i,L)}\right)) \tag{5}$$

We use each i^{th} data point in X as $\mathbf{C}^{(i,1)}$ for layer-1. There are K_l convolutional kernels $\mathbf{F}^{(k_l,l)}$ for every layer-l. Each of the kernels computes a 2D feature map $\tilde{\mathbf{C}}^{(i,k_l,l)}$. All the 2D feature maps in the same layer are stacked with spatial alignment before they can be passed to the following pooling layer. The pooling layer exports a 3D tensor $\mathbf{C}^{(i,l)}$ using max pooling strategy with stride being 2 and kernel size being 2 by 2. The ReLU function is used as the activator $h(\cdot)$. For multi-channel images, each channel is processed individually and stacked in the Conv Layer.

3.2 DGDN based face decoder

The decoder in this paper is achieved with a deep generative deconvolutional network. There are two main types of layers in this network, which are the up-sampling layers and the convolutional layers. As similar in the encoder, those two types of layers process the data in an alternating way:

$$\text{Up-sampling Layer:} \quad \mathbf{G}^{(i,l-1)} \sim \text{upsizing}\left[\widetilde{\mathbf{G}}^{(i,l-2)}\right] \tag{6}$$

$$\text{Conv Layer:} \quad \widetilde{\mathbf{G}}^{(i,l-1)} = \sum\left\{h\left(\mathbf{G}^{(i,k_{l-1},l-2)} * \mathbf{D}^{(k_{l-1},l-1)}\right)\right\}_{k_{l-1}=1}^{K_{l-1}} \tag{7}$$

$$\text{Up-sampling Layer:} \quad \mathbf{G}^{(i,l)} \sim \text{upsizing}\left[\widetilde{\mathbf{G}}^{(i,l-1)}\right] \tag{8}$$

$$\text{Conv Layer:} \quad \widetilde{\mathbf{G}}^{(i,l)} = \sum\left\{h\left(\mathbf{G}^{(i,k_l,l)} * \mathbf{D}^{(k_l,l)}\right)\right\}_{k_l=1}^{K_l} \tag{9}$$

$$\text{Reconstruction:} \quad X = \left\{x^{(i)} \sim P\left(\widetilde{\mathbf{G}}^{(i,l)}, \frac{\mathrm{I}}{\alpha_0}\right)\right\}_{i=1}^{N} \tag{10}$$

As shown in Fig. 2, the up-sampling layers and the convolutional layers together achieve the deconvolution from the encoded data towards the final reconstructed image. There are multiple ways to up-sample the data from the previous layer. One is to pad new pixel values outside the original image boundary when the size of the original image is small, as shown in the Fig. 2. We can also up-sample the original image by filling the interpolated pixels with

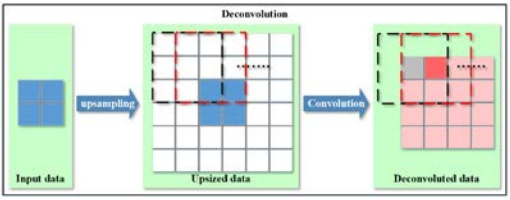

Figure 2: The illustration for deconvolution process. The up-sampling process is achieved by padding pixels outside the image boundary for 2x2 input data, and by nearest-neighbour interpolation for larger sized input data.

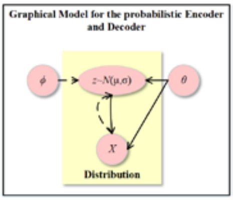

Figure 3: The graphical model for the probabilistic relationship between the encoding and the decoding process. ϕ and θ are the parameters for the encoder and the decoder respectively.

a nearest-neighbor strategy in a pre-defined stride (i.e., 2) after the first layer. $\mathbf{G}^{(i,l)}$ can be treated as a 3D tensor with K_l slices for the layer-l of the decoder network, where $l \in \{1, \dots, \tilde{L}\}$. It is actually the latent code (feature map) for the encoded image when in layer-1. The feature map $\mathbf{G}^{(i,l)}$ is convolved by 3D convolutional kernel $\{\mathbf{D}^{(k_l,l)}\}_{k_l=1}^{K_l}$, and all the slices are summed up before being passed to the next layer. ReLU is still used as the activator $h(\cdot)$. In the final layer, the output data will have a distribution approximating $\mathbb{E}(X)$ with the precision being α_0 to achieve image reconstruction.

4 Model parameter learning

The mathematical basis of the variational autoencoders is actually to solve a distribution estimation problem as shown in Eq. (11),

$$P(X) = \int P_\theta(X|z) Q_\phi(z) dz \tag{11}$$

where $P_\theta(X|z) = \mathcal{N}(X|f(z;\theta), \sigma^2 * I)$ allows us to make the dependence of the datapoints X on the latent vector z, and the decoder parameter θ. $Q_\phi(z)$ indicates a certain probability density function, i.e., $\mathcal{N}(0, I)$, to sample the latent vector z from a high-dimensional space $\mathcal{Z}$, where $z \in \mathcal{Z}$.To make sure based on this distribution the sampling of the z will properly produce a similar X, we use $Q_\phi(z|X)$ to describe this distribution depending on the data X explicitly. This is a process conducted by the encoder with the model parameter of ϕ. Fig. 3 is a graphical model showing this relationship between the encoding and decoding processes.

To design the loss function for the model training, we use $Q_\phi(z|X)$ to describe the distribution to generate latent vector z given the training data X in the encoder. We also use $P_\theta(X|z)$ to describe the distribution to reconstruct X given z in the decoder. The parameters ϕ and θ need to be optimized during the training process. The loss function has two parts: a distance between the reconstructed data $\hat{X}$ and training data X; a distance between the model z sampling distribution $Q_\phi(z|X) = \mathcal{N}(z|\mu(X;\vartheta), \sigma(X;\vartheta))$ and the target z sampling distribution, which is $P(z) = \mathcal{N}(z|0, I)$ for most cases.

$$\begin{aligned} \mathrm{Loss}_{\mathrm{xent}} &= H\big[P(X), \mathbb{E}_{z\sim Q}[P_\theta(X|z)]\big] = \mathbb{E}\big[-P(X)\log \mathbb{E}_{z\sim Q}[P_\theta(X|z)]\big] \\ &= -\textstyle\sum_{i=1}^{N}[x_i \log \hat{x}_i + (1-x_i)\log(1-\hat{x}_i)] \end{aligned} \tag{12}$$

$$\begin{aligned} \mathrm{Loss}_{D_{KL}} &= \mathcal{D}[P(z)||Q_\phi(z|X)] = \mathbb{E}_{z\sim Q}\big[\log P(z) - \log Q_\phi(z|X)\big] \\ &= -\tfrac{1}{2}(1 + \log\sigma^2 - \mu^2 - e^{\sigma^2}) \end{aligned} \tag{13}$$

$$\mathrm{Loss} = \mathrm{Loss}_{\mathrm{xent}} + \mathrm{Loss}_{D_{KL}} \tag{14}$$

Hence, the total loss consists of a cross entropy $\text{Loss}_{\text{xent}}$ and a Kullback-Leibler divergence $\text{Loss}_{D_{KL}}$ as shown from Eq. (12) to (14). In the training process, the proposed model uses the manually crafted symmetric faces as the training data. Therefore, the size of the training data cannot be large enough to train the whole model. To tackle this, we design a multi-stage training process for the task.

4.1 A multi-stage training strategy

The objective of the proposed model is to generate a symmetric face when given a normal facial image. However, in practice, there is no available database dedicated for such symmetric face to support end-to-end model training process. To tackle this problem, we design a multi-stage model training process to avoid the demand for a large amount of symmetric face as training data.

It is obvious that the generating of the symmetric face is highly related to the reproduction of the facial image, as well as the extraction of the facial region. Therefore, as shown in Fig. 1, to establish the proposed model, we can firstly train the model to reconstruct the facial image. There are sufficient facial images enough as the training data publicly available. And then in the second stage, the model is re-trained to generate face only images. And in the final stage, the model is transferred for the symmetric face generating task using the manually prepared training data. This strategy can make the training data of symmetric face to be substantially less than the training data used for the first stage.

5 Experiments

5.1 Dataset

The proposed facial symmetrization model is trained on the *Labelled Faces in the Wild* (LFW) facial database [31] of more than 13K unconstrained facial images collected from the web. Those data include the facial images with different head poses, different ambient illuminations, different face sizes, and some of them with different facial occlusions such as shades or hands. The facial images are taken from the real photographs in the wild, which covers different skin colours, genders and ages. The only constraint of those facial images is that they are detected by the Viola-Jones face detector [32].

This database provides facial labels, yet only the facial images are utilized to train the proposed model. We use 10K images as the training data for the first training stage,1K images for validation, and 2K images for the evaluation of the generating performance of the proposed model. We also manually crop facial regions for 5K of the facial images for the second training stage. The facial middle lines on those face-only images are manually marked as well to generate 5K symmetric faces for the third training stage. To further test the performance of the proposed method, we also conducted evaluations on the 2500 2D facial images from the database of *Binghamton University 3D Facial Expression* (BU-3DFE) [33]. The training process is conducted based on a GTX1070 GPU.

5.2 Symmetric face generating

We evaluate our method on the face database of LFW and BU3D, which respectively provide the uncontrolled and controlled facial images. The visual examples of the experiment results are demonstrated in Fig. 4.

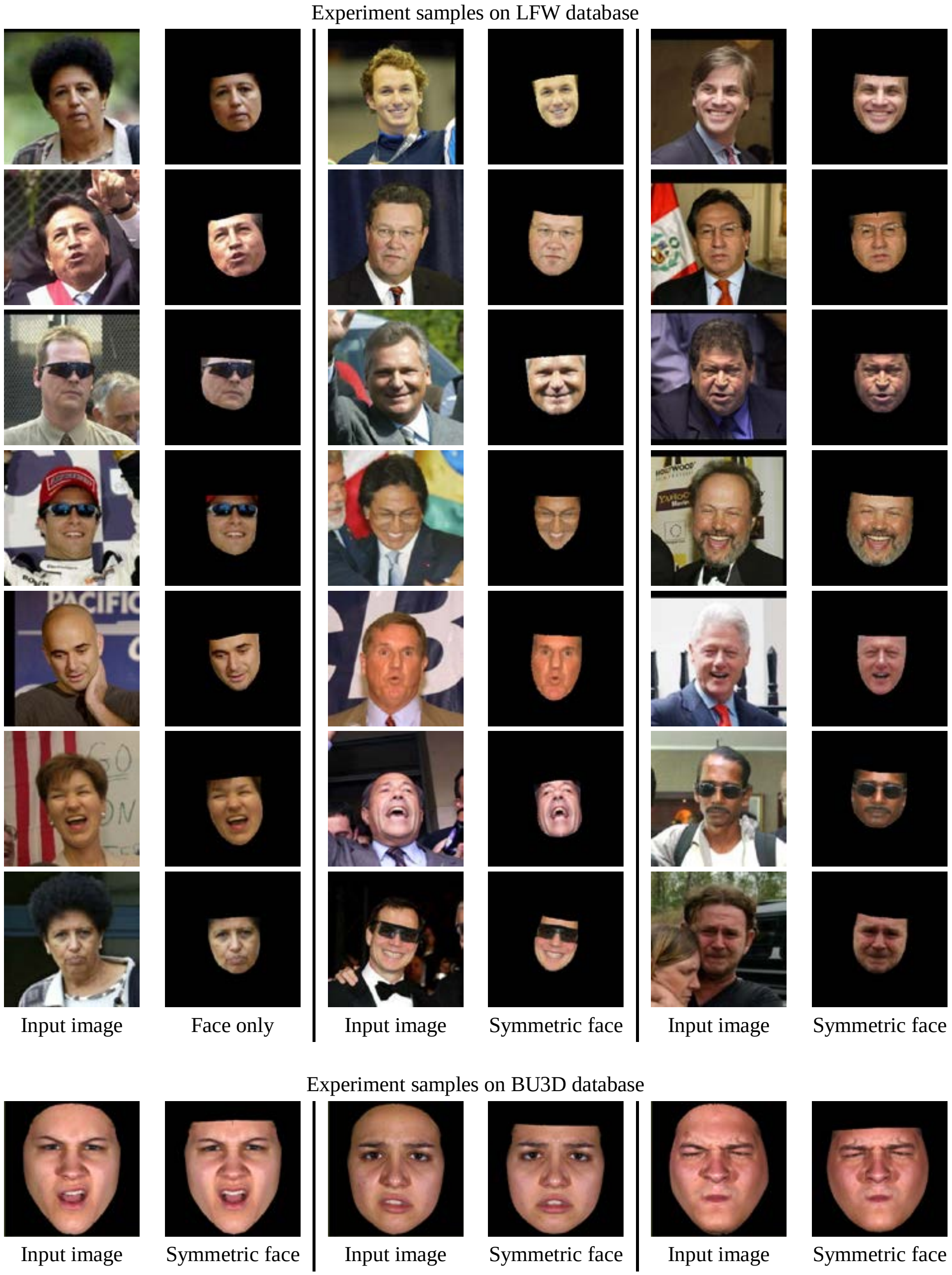

Figure 4: Samples of the experiment results for symmetric face generating on both LFW and BU3D database. Since the proposed model is trained in three stages, some generated face only images are also demonstrated. It can be noticed that some test examples with large head poses or facial occlusions such as shades, can also be well handled by the proposed model. Those will pose more challenges for traditional facial landmark based methods.

	Ours	CLNF [28]	SDM [22]	CLM [27]	Dlib [34]
Recognition Accuracy of Symmetrized Faces with respect to the Original Faces	**0.872**	0.738	0.694	0.658	0.712

Table 1: The performance comparisons of the proposed method with several most popular approaches

It can be observed in Fig. 4 that the generated symmetric faces look very similar to the original input facial images, except that they are perfectly symmetric. For the uncontrolled facial images in LFW database, it can be noticed that there are test data with large head poses, as well as the facial occlusions caused by the shades or the hands. Those conditions can pose much more challenges for the traditional symmetric face generating methods that based on the facial landmark detections. On the other hand, the proposed method analyses the whole facial images to extract the facial region and generate a symmetric face, which makes the proposed method more robust to above conditions. The noises on the local regions have less impacts on the performance of our method. Since the proposed model is trained based on a 3-stage strategy, we also present some experiment results for the extracted face-only images using the model trained after second stage.

Beside those visual demonstrations, we also compare our method with several other traditional methods based on different facial landmark detection algorithms, including some of the most robust and popular ones such as CLNF [28], SDM [22], CLM [27] and Dlib [34]. The generated symmetric faces have the same facial identities of the original images, based on which we employ facial recognition algorithm to identify whether the generated symmetric face belongs to the same person in the original image. This yields a recognition accuracy that can reflect the symmetric face generating performance. The performance comparisons based on this quantitative measurement are demonstrated in Table 1. The *OpenFace* [35] is utilized to achieve the facial recognition in these evaluation experiments due to its excellent recognition performance on LFW database [36]. As it can be seen, the performance using the traditional facial landmark based methods can be negatively impacted, especially when finding facial middle line under uncontrolled conditions even those traditional methods have a good performance on facial landmark detection task. The proposed method can achieve a competitive and robust result by the full image analysis.

6 Conclusion

Based on the structure of the variational autoencoders, this paper presents a generative model for symmetric face reconstruction given an unconstrained facial image. The proposed model consists of a convolutional neural network as the encoder, and a deep generative neural network as the decoder. The generative model in this paper utilizes a 3-stage coarse-to-fine training strategy, which avoids the demand for the large amount of the symmetric face as training data. In the first stage, the generative model is trained to reproduce the whole facial images. The training data for this stage is sufficient enough since there are plenty of facial image database publicly available. In the second stage, the model is re-trained to extract the face only images from the input data. In the final stage, the model is transferred to generate symmetric faces based on the manually crafted training data, which can be substantially less that what required in the first training stage. The performance of the proposed model is demonstrated in the experiments. Along with the visual demonstration, the proposed method also achieves competitive performance compared with several other solutions based on some of the most popular facial landmark detection methods in this field.

Acknowledgement

This work was supported by the Project of Shandong Province Higher Educational Science and Technology Program (No. J17KA076), the Scientific Research Foundation of Shandong University of Science and Technology for Recruited Talents (No. 2017RCJJ047), and National Natural Science Foundation of China (NSFC) (No. U1706218).

References

[1] T. Wang, S. Zhang, J. Dong, L. Liu, and H. Yu, "Automatic evaluation of the degree of facial nerve paralysis," *Multimed. Tools Appl.*, vol. 75, no. 19, pp. 11893–11908, 2016.

[2] T. Wang, J. Dong, X. Sun, S. Zhang, and S. Wang, "Automatic recognition of facial movement for paralyzed face," *Biomed. Mater. Eng.*, vol. 24, no. 6, pp. 2751–2760, 2014.

[3] M. Sajid, T. Shafique, M. J. A. Baig, I. Riaz, S. Amin, and S. Manzoor, "Automatic Grading of Palsy Using Asymmetrical Facial Features: A Study Complemented by New Solutions," *Symmetry (Basel).*, vol. 10, no. 7, p. 242, 2018.

[4] K. Fujiwara, Y. Furuta, N. Yamamoto, K. Katoh, and S. Fukuda, "Factors affecting the effect of physical rehabilitation therapy for synkinesis as a sequela to facial nerve palsy," *Auris Nasus Larynx*, vol. 45, no. 4, pp. 732–739, 2018.

[5] A. Song, G. Xu, X. Ding, J. Song, G. Xu, and W. Zhang, "Assessment for facial nerve paralysis based on facial asymmetry," *Australas. Phys. Eng. Sci. Med.*, vol. 40, no. 4, pp. 851–860, 2017.

[6] H. Cui, Y. Chen, W. Zhong, H. Yu, *et al.*, "The asymmetric facial skin perfusion distribution of Bell's palsy discovered by laser speckle imaging technology," *Clin. Hemorheol. Microcirc.*, vol. 62, no. 1, pp. 89–97, 2016.

[7] C. Saegusa, J. Intoy, and S. Shimojo, "Visual attractiveness is leaky: the asymmetrical relationship between face and hair ," *Frontiers in Psychology* , vol. 6. p. 377, 2015.

[8] G. Rhodes, F. Proffitt, J. M. Grady, and A. Sumich, "Facial symmetry and the perception of beauty," *Psychon. Bull. Rev.*, vol. 5, no. 4, pp. 659–669, 1998.

[9] G. Zhang and Y. Wang, "Asymmetry-based quality assessment of face images," in *International Symposium on Visual Computing*, 2009, pp. 499–508.

[10] D. I. Perrett, D. M. Burt, I. S. Penton-Voak, K. J. Lee, D. A. Rowland, and R. Edwards, "Symmetry and human facial attractiveness," *Evol. Hum. Behav.*, vol. 20, no. 5, pp. 295–307, 1999.

[11] Y. Wang, H. Yu, J. Dong, M. Jian, and H. Liu, "Cascade Support Vector Regression-based Facial Expression-Aware Face Frontalization," *2017 IEEE International Conference on Image Processing*. 2017.

[12] T. Hassner, S. Harel, E. Paz, and R. Enbar, "Effective face frontalization in unconstrained images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4295–4304.

[13] B. R. Mallikarjun, V. Chari, and C. V Jawahar, "Efficient face frontalization in unconstrained images," in *Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), 2015 Fifth National Conference on*, 2015, pp. 1–4.

[14] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker, "Towards Large-Pose Face Frontalization in the Wild," in *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[15] C. Sagonas, Y. Panagakis, S. Zafeiriou, and M. Pantic, "Robust statistical face frontalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3871–3879.

[16] G. Sikander, S. Anwar, and Y. A. Djawad, "Facial Feature Detection: A Facial Symmetry Approach," in *2017 5th International Symposium on Computational and Business Intelligence (ISCBI)*, 2017, pp. 26–31.

[17] Y. Xu, Z. Zhang, G. Lu, and J. Yang, "Approximately symmetrical face images for image preprocessing in face recognition and sparse representation based classification," *Pattern Recognit.*, vol. 54, pp. 68–82, 2016.

[18] H. Yu and H. Liu, "Regression-based facial expression optimization," *IEEE Trans. Hum. Mach. Syst.*, vol. 44, no. 3, pp. 386–394, 2014.

[19] H. Salam and R. Séguier, "A survey on face modeling: building a bridge between face analysis and synthesis," *Vis. Comput.*, vol. 34, no. 2, pp. 289–319, 2018.

[20] A. Al-Rahayfeh and M. Faezipour, "Eye Tracking and Head Movement Detection: A State-of-Art Survey," *IEEE J. Transl. Eng. Heal. Med.*, vol. 1, p. 2100212, 2013.

[21] S. Zhang, H. Yu, T. Wang, L. Qi, J. Dong, and H. Liu, "Dense 3D facial reconstruction from a single depth image in unconstrained environment," *Virtual Real.*, vol. 22, no. 1, pp. 37–46, 2018.

[22] X. Xiong and F. De la Torre, "Supervised descent method and its applications to face alignment," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 2013, pp. 532–539.

[23] I. Masi, S. Rawls, G. Medioni, and P. Natarajan, "Pose-Aware Face Recognition in the Wild," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[24] J. Harguess and J. K. Aggarwal, "Is there a connection between face symmetry and face recognition?," in *CVPR 2011 WORKSHOPS*, 2011, pp. 66–73.

[25] J. Harguess and J. K. Aggarwal, "A case for the average-half-face in 2D and 3D for face recognition," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2009, pp. 7–12.

[26] I. Matthews and S. Baker, "Active appearance models revisited," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 135–164, 2004.

[27] D. Cristinacce and T. Cootes, "Automatic feature localisation with constrained local models," *Pattern Recognit.*, vol. 41, no. 10, pp. 3054–3067, 2008.

[28] T. Baltrusaitis, P. Robinson, and L.-P. Morency, "Constrained Local Neural Fields for Robust Facial Landmark Detection in the Wild," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, 2013.

[29] G. Passalis, P. Perakis, T. Theoharis, and I. A. Kakadiaris, "Using facial symmetry to handle pose variations in real-world 3D face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 10, pp. 1938–1951, 2011.

[30] C. Doersch, "Tutorial on Variational Autoencoders," *ArXiv e-prints*, Jun. 2016.

[31] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua, "Labeled Faces in the Wild: A Survey," in *Advances in Face Detection and Facial Image Analysis*, M. Kawulok, M. E. Celebi, and B. Smolka, Eds. Cham: Springer International Publishing, 2016, pp. 189–248.

[32] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2004.

[33] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato, "A 3D facial expression database for facial behavior research," in *7th International Conference on Automatic Face and Gesture Recognition (FGR06)*, 2006, pp. 211–216.

[34] D. E. King, "Dlib-ml: A Machine Learning Toolkit," *J. Mach. Learn. Res.*, vol. 10, pp. 1755–1758, 2009.

[35] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, "OpenFace: A general-purpose face recognition library with mobile applications," 2016.

[36] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.