Effect of instance normalization on fine-grained control in sketch-based face generalization

Anonymous cvm submission

Paper ID 196

Abstract

In this paper we investigate the effect of instance normalization on fine-grained control in generating photorealistic face images from hand-drawn sketches. Exiting image-to-image translation methods mostly utilize instance normalization after activation as nomalization layer. But this induces the problem of missing some precise textures in the generated images and weakening face editing performance as we modify the input sketches. To address this issue, we improve upon baseline model on the network architecture. We show how a small change in the generator architecture results in a significant improvement of the image generation quality. User studies demonstrate that our method surpass the baseline method regarding both face editing and image fidelity.

1. Introduction

Photorealistic face image synthesis from hand-drawn sketches has drawn a lot of attention in computer graphics and computer vision for many years. The typical approaches use generative and adversarial networks(GANs)[4], whose generator is built by stacking convolution, normalization, and nonlinearity layers, as their network architectures. Normalization layers, in fact, can normalize the parameter distribution so as to alleviate the issue of slow convergence in gradient update process and avoid gradient vanishing and gradient exploding, which is vastly important in GANs.

There are many normalization means with different goals, such as batch normalization[12], group normalization[18], layer normalization[10] and instance normalization[16]. Batch normalization eliminates the influence of internal covariate shift, effectively avoids the possible problems of gradient vanishing and gradient exploding in the process of gradient backpropagation, and speeds up the training time. Group normalization organizes the channels of a layer into different groups, and computes the mean and standard deviation within each group independently for normalization. It is independent of batch size, thus it's frequently used in tasks which prefers small mini-batch size, such as object detection and video classification. Layer normalization computes the mean and variance used for normalization over all the channels of a single layer. It's more suitable to apply it to recurrent neural networks. Unlike batch normalization, layer normalization performs exactly the same computation at training and test times.

Instance normalization is similar to layer normalization but goes one step further: it computes the mean and variance for normalization over each channel in each training example. Recent studies show that instance normalization performs well on visual tasks such as style transfer and image translation[17, 13, 8] when replacing batch normalization in GANs architecture. Nonetheless, instance normalization layers also tend to "wash away" information contained in the input sketches, thus it results in descent of the feature expression ability and imprecise control on detailed textures generation.

It is important to support fine-grained control in sketch-based content creation. So we investigate the effect of instance normalization on fine-grained control in sketch-based photorealistic face image generation using data reduction and visualization methods. We utilize PCA(principal component analysis)[2] to visualize and analyze features extracted by the generator from sketches. Consequently, we propose to remove the first two instanse normalization layers in the baseline generator, and we show that this change in the generator architecture results in a significant improvement to control accuracy in image generation. We conduct experiments and interactive user studies to evaluate our proposed method, and the results demonstrate that our method surpass baseline method on control performance on the sketch-to-image task.

2. Related Work

2.1. Image-to-Image Translation

The goal of image-to-image translation task is to convert the input images from one domain to another given the input-output image pairs as training data, in other words, to generate corresponding image according to the input image and meanwhile the two images share the same structure and scene. At present, many researchers employ adversarial manner to train deep neural networks in image-to-image translation tasks[8, 19, 3, 6, 1, 14, 5, 14, 9].

The concept of image-to-image translation was first proposed by pix2pix[7], which is based on generative adversarial networks conditioned on images. The network architecture of pix2pix is composed of generator G and discriminator D, the former is responsible for converting the input image from source domain to target domain, and the latter is responsible for telling the generated images apart from the real images. This model can be applied to a variety of different image translation scenarios, such as lable maps to streetscapes, edge maps to photos, image colorization and so on. However, the drawback of pix2pix is also obvious. At most, it can only generate images with a resolution of 256 $\times$ 256. If pix2pix is forced to generate images with a higher resolution, the training process will be unstable and the generation quality will decline. Therefore, Ting-Chun Wang et al. proposed a new image-to-image translation model referred to as pix2pixHD[17], which is based on pix2pix, aiming to improve the resolution of generated images from semantic lable maps. It adopts a coarse-to-fine generator and a multiscale discriminator. And it can also be applied to edge-to-photo generation conditioned on edge map and photo pairs. However, the large gap between synthesized edge maps and hand-drawn sketches challenges the generalization ability of these models. In order to efficiently preserve and propagate semantic information throughout the network, GauGAN[13] which can effectively turn doodles into reality, utilizes semantic segmentation masks to modulate the activations in normalization layers through a spatially-adaptive, learned transformation. It inspires us to investigate the effect of normalization layers on information propagation in network architecture.

2.2. Normalization Layers

Normalization layers have been an important component in modern deep neural networks for stabilizing the training process. The following are several common normalization methods. Batch normalization[12] is a method that normalizes activations in a network across the mini-batch. It calculates the mean and variance among one channel over each mini-batch. Then, it learns two parameters to scale and shift the normalized activations. Batch Normalization provides a really strong way to reduce internal covariant shift problem and can also stabilize the neural networks to speed up training process. Group normalization[18], as its name suggests, divides the channels of activations into groups and then calculates the mean and standard deviation over the group of channels of each training sample for normalization, which is frequently adopted in some tasks such as object detection, semantic segmentation and video classification. It helps deep learning model work better at small mini-batch size. Layer normalization[10] computes the mean and variance used for normalization over all the channels of a single layer. It's more suitable to apply it to recurrent neural networks. Unlike batch normalization, layer normalization performs exactly the same computation at training and test times. In fact, instance normalization[16] is very similar to batch normalization. The only difference is that batch normalization computes the mean and variance among a mimi-batch, while instance normalization only operates across each channel in each training sample. It has a remarkable effect on tasks such as style transfer and image translation. It is used in the baseline method[17] and other image translation tasks[13, 8, 19, 9].

3. Methods

3.1. The pix2pixHD Baseline

Pix2pixHD[17] is an image translation model based on conditional generative adversarial network, which can generate high-quality and high-resolution images from input semantic label maps. It adopts a improved adversarial loss, as to the network architecture it introduces a coarse-to-fine generator and a multiscale discriminator. Using this model, a more realistic image with a resolution of 2048 $\times$ 1024 can be generated, which is better than the previous method. In addition, the model can also be used for interactive image editing. First of all, we can add the instance segmentation information of the object into the input to realize the editing of the object, such as adding or deleting objects or changing the category of objects in the generated image. Second, you can edit the appearance of an object in the generated image given the same input.

Its generator is divided into two sub networks: G_1 and G_2. G_1 is alled global generator network, and G_2 is called local enhancer network, where G_1 is used to generate base image, and G_2 is used to improve the resolution of the image. In order to distinguish real and

generated images with high resolution, the discriminator needs to have a large perception field. Therefore, the model uses a multi-scale discriminator which can preserve both global and local information. The loss function of this model is composed of three parts: adversarial loss $L_{GAN}(G, D_k)$, feature matching loss $L_{FM}(G, D_k)$ and VGG perceptual loss $L_{VGG}(G)$. The full objective is formulated as:

$$\min_G \left(\left(\max_{D_1,D_2,D_3} \sum_{k=1,2,3} L_{GAN}(G, D_k) \right) + \lambda \left(\frac{1}{3} \sum_{k=1,2,3} L_{FM}(G, D_k) + L_{VGG}(G(\boldsymbol{s})) \right) \right) . \quad (1)$$

where λ controls the importance of the three terms.

3.2. Network Architecture

In generator, the activation of convolutional output is normalized in the channel-wise manner and then modulated with unified scale and bias within channel. This operation, to a certain extent, leads to a negative effect that a local change in the input sketch broadcasts globally, thus resulting in a degraded capacity of fine-grained control on generated image.

However, the vanishing gradient or exploding gradient problem is bound to emerge when the instance normalization layers in feature extraction stage of the generator network is abandoned too much, which makes the training process difficult to converge. Based on the consideration above, we remove the first two instance normalization layers in the global generator of pix2pixHD baseline and take the rest as our own generator. The architecture of our generator is shown in Fig. 1 . Our generator operates at a resolution of 512×512. It consists of 4 components: a convolutional front-end without normalization $G_1^{(F)}$, a downsampled convolutional mid-end $G_1^{(M)}$, a set of residual blocks $G_1^{(R)}$ and a transposed convolutional back-end $G_1^{(B)}$. $G_1^{(F)}$ composed of two unnormalized convolution and activation layers can remain underlying information of input sketches locally, scilicet information flowing through the network without broadcasting. Hence efficient fine-grained control on generated images can be conduct.

The discriminator applys the multi-scale design consistent with pix2pixHD. There are 3 discriminators that have an identical network structure but operate at different image scales. In addition, each discriminator is built on PatchGAN architecture proposed by Isola Phillip et al.[7]. Sketches at different scales are concatenated with corresponding face images, and fed into the discriminators, respectively.

4. Experiment

We conduct extensive experiments to evaluate our proposed method. In Sec. 4.1, we visualize the feature vectors extrcted from hand-drawn sketches by the generator of pix2pixHD baseline and our model, and analyze the visualization results to investigate the effect of instance normalization on fine grained control in generated face images. In Sec. 4.2, we introduce how to produce our training and testing datasets. Next we introduces the method of data augmentation aiming to alleviate the problem that the generated image has poor tolerance to the spatial position change of the input sketches, and shows the generation results of the model trained with augmented data. Further more, we conduct comparative experiments on hand-drawn sketches between our methods and pix2pixHD baseline.

4.1. Feature Visualization

We extract the feature vector of the middle layers of the generator network and analyze them using two data visualization tools comparatively.

Feature vector extraction In order to verify whether the existing image translation neural network can extract the face shape features consistent with the user's intention for the hand-drawn sketches with low accuracy and geometric deformation, we draw 198 sketches of resolution 512 × 512. Fig. 2 shows examples of these 11 sets of hand-drawn sketches.

We refer to the first 5 layers of our generator as $L_0, \ldots, L_4$. With the hand-drawn sketches fed into the generator, we get 5 feature maps as output of L_0 to L_4, and then we extract one full channel feature vector on the left eye corner point of coordinate (170, 250) from each of the 5 feature maps respectively. We call the 5 vectors $\boldsymbol{v}_k$, $k = 0, \ldots, 4$. The coordinates of the corresponding points on the five feature maps are (170, 250), (85, 125), (43, 63), (22, 32) and (11, 16). And the perceptive fields of the 5 vectors are 7×7, 9×9,13×13, 21×21 and 37×37 as shown in Fig. 3. The dimensions of these 5 feature vectors are 48, 96, 192, 384 and 768, respectively. We get the 5 feature vectors of pix2pixHD generator called $\boldsymbol{v}_0', \ldots, \boldsymbol{v}_4'$ in the same way.

Visualization with PCA Principal Component Analysis (PCA)[2] is a common linear data dimension reduction method, which can better retain the statistic characteristics of data in high dimensional space. We use PCA to perform visually analysis on $\boldsymbol{v}_0 \sim \boldsymbol{v}_4$ after dimension reduction. Fig. 4(a) and Fig. 4(b) demonstrate the results of PCA visualization on $\boldsymbol{v}_0$ and $\boldsymbol{v}_1$, and Fig. 4(c) is a legend which is applicable to the following chapters. The corresponding relationship between hand-drawn sketch categories and number colors are: G1 (light blue), G2 (light purple), G3

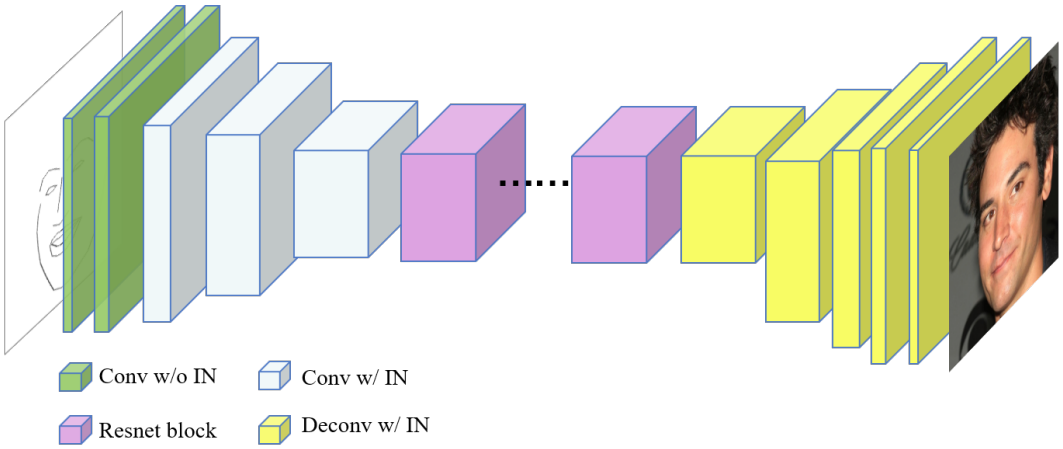


Figure 1. Network architecture of our generator.

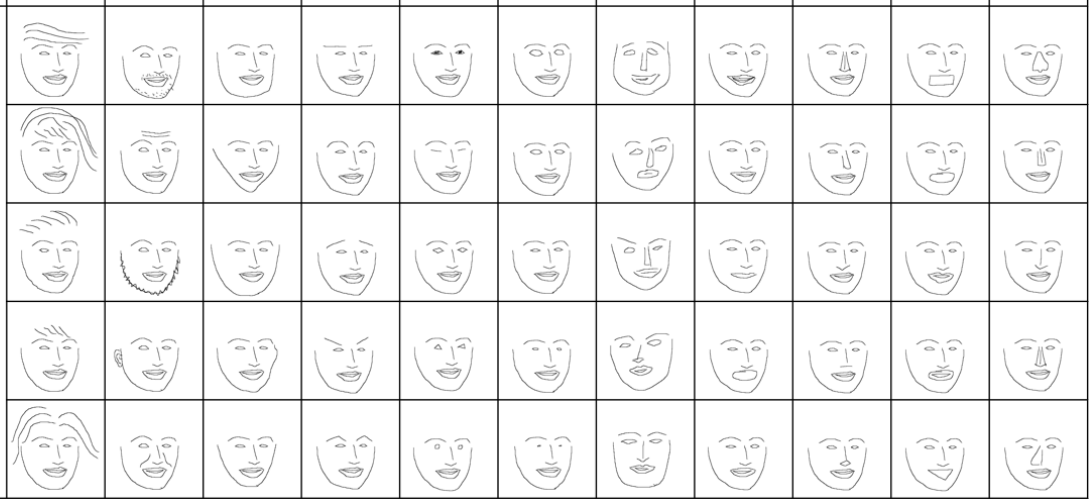
class	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11
amount	26	15	12	17	19	9	12	26	19	26	19
Examples of hand-drawn sketches											

Figure 2. These sketches can be divided into 11 categories. G1:Add hair; G2:Add new attributes, such as whiskers, wrinkles, ears; G3:Change face shape; G4:Change eyebrows; G5:Change eye shape; G6:changes eye size; G7:Graffiti-drawn; G8:Change mouth; G9:Change nose; G10:Change mouth(same eyes as G9); G11:Change nose(same eyes as G8). Specifically, there is no correlation between G7 and the other 10 classes. Except G7, the sketch only changes a particular location or property while the rest of the sketch remains the same. Except for the same eye between G8 and G11, G9 and G10, the eye lines of other categories of sketches are different.

(pale red), G4 (blue), G5 (orange), G6 (olive green), G7 (pink), G8 (gray), G9 (purple), G10 (light green), G11 (yellow). It can be seen from the figure that the data of G1, G2, G3 and G4 are distributed at a same point respectively, the data of G8 and G11 are gathered on the same point, the data of G9 and G10 are distributed at the same point, whereas the data of G5, G6 and G7 are distributed dispersedly. This indicates that after the removal of the instance normalization in the first two layers, the features of the left eye corner point are only affected by the contents of the sketch in the corresponding perceptive field, and the change of the sketch in the perceptive field will change the extracted features of the corresponding points but will not beyond the perceptive field. Visualization results of $\boldsymbol{v}_2 \sim \boldsymbol{v}_4$ are shown in

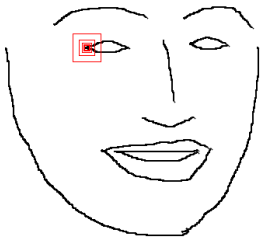


Figure 3. The perceptive fields of corresponding left eye corner points in the feature maps $L_0 \sim L_4$.

Fig.5(a)(b)(c). By observing visualization results of $\boldsymbol{v}_2$, we can find out that the feature vectors belonging to categories with unmodified eyes have a different spatial distribution within class, which indicates that instance normalization conveys modification in other regions to left eye corner of sketches resulting in an effect on features extracted from left eye corner. By comparing visualization results of $\boldsymbol{v}_2$ and $\boldsymbol{v}_4$, we can find out that the feature vectors belonging to categories with unmodified eyes such as G1, G2, G3, G4, G8, G9, G10, G11, distribute more dispersedly within class, which indicates that with the instance normalization stacking layer by layer, changing other parts on the input sketch has more and more obvious influence on the features of the eye position, leading to the changes of the eyes in the generated image.

We also contrast the PCA visualization results of $\boldsymbol{v}_2 \sim \boldsymbol{v}_4$ and $\boldsymbol{v}_2^{'} \sim \boldsymbol{v}_4^{'}$ respectively. Fig.6(a)(b)(c) illustrate the visualization results of $\boldsymbol{v}_2 \sim \boldsymbol{v}_4$, and Fig.6(d)(e)(f) illustrate the visualization results of $\boldsymbol{v}_2^{'} \sim \boldsymbol{v}_4^{'}$. The $\boldsymbol{v}_2^{'}, \boldsymbol{v}_3^{'}, \boldsymbol{v}_4^{'}$ belonging to categories with unmodified eyes such as G1, G2, G3, G4, G8, G9, G10, G11, have a bigger in-class distance compared with $\boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4$. This exactly shows the removal of the first two instance normalization layers in pix2pixHD generator can better extract the low-level features of the input sketch, which can effectively reduce the influence of changing one part of the sketch on other parts of the generated image, and enhance fine grained control on the generated image.

4.2. Results

To explore the IN on fine grained control in generated images, we perform experiments on our model with hand-drawn sketches. We first report the dataset synthesis methods where datasets contain contours for training and hand-drawn sketches for testing. Then we introduce data augmentation aiming for addressing the issue that the generated image has poor tolerance to the spatial change of the input sketch. At last, we provide qualitative comparison against baseline method and show that our method achieves a better fine-grained control on generated images.

Datasets CelebA-HQ dataset contains more than 30,000 face images generated by PGGAN[15] of resolution 1024×1024. Since the images are cropped according to face landmarks, each image is global aligned. We extracted 68 landmarks from the face images in the CelebA-HQ dataset, and then connected these points in sequence with a line of pixel 2 to form the contour. As the face photos in Celeba-HQ dataset are global aligned, so are the contours. Contour is more simple and clean than other types of sketch like edge maps[11] and mask edge maps[9], so it is more suitble to imitate human hand-drawn sketch. Therefore, it is more reasonable to use contour as training data. For the purpose of the experiment, we scaled the picture of resolution 512×512. After selection, the training set contains 14,973 pairs of contour and face photos, while the test set contains 4,992 pairs of contour and face photos. In order to evaluate the generalization ability of our model on the hand-drawn sketches, we develop an interactive interface for drawing sketches and displaying the generated photos in real time.

Data augmentation Since the face photos of the CalebA-HQ dataset are cropped with the reference of facial landmarks, and our training data contours are also obtained based on facial landmarks, we intuitively believe that all the facial landmarks of the training data have spatial consistency. To verify the hypothesis above, we calculate the average face of all the training data, as shown in Fig.7. We find that the facial features and facial contours of training data are basically in the same position, in other words, the training data is global aligned. This issue leads to degraded generalization ability of the model. In order to imitate the human hand-drawn sketches, we apply random translation and rotation to the training contours. Specifically, offsets randomly selected from $[-d, d]^2$ and angles randomly selected from $[-\theta, \theta]$ are added to the training contours, where d is the maximum offset and θ is the maximum angle and we set $d = 25$, $\theta = 7^\circ$ in our experiments. But face photos are not translated or rotated because we expect the generated images to remain global aligned regardless of the spatial location of the input sketches. Fig.8 illustrates the comparisons between images generated before data augmentation and that after data augmentation.

Qualitatively comparisons As you can see in Fig.9, our model generates the most photorealistic images in contrast to the M_2 model producing images with lowest sense of reality. It indicates that removing too many instance normalization layers in generator can weaken the model as reason of gradient vanishing.

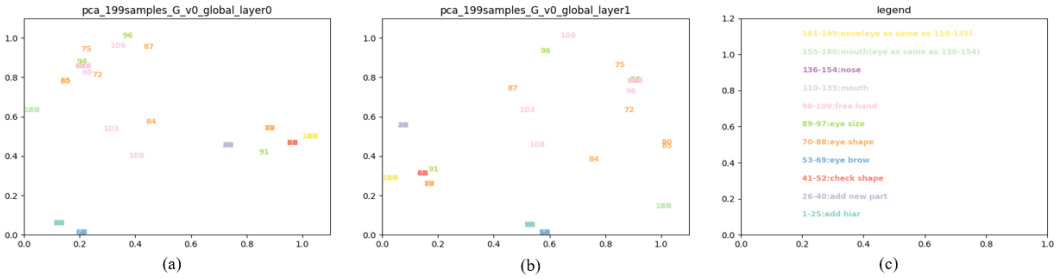


Figure 4. Results of PCA visualization on $\boldsymbol{v}_0$ and $\boldsymbol{v}_1$

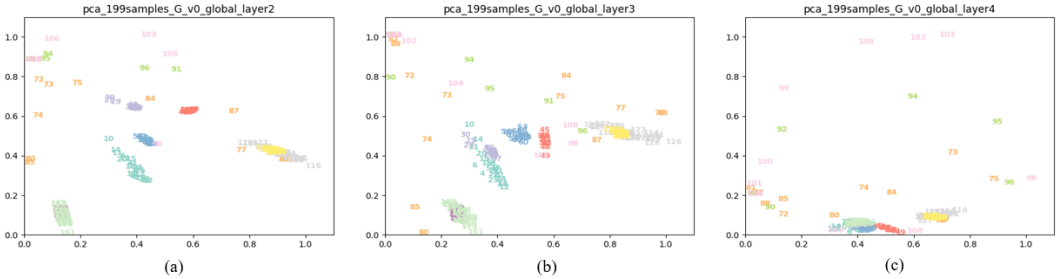


Figure 5. Visualization results of $\boldsymbol{v}_2 \sim \boldsymbol{v}_4$

We perform comparative experiments between our model and pix2pixHD baseline on hand-drawn sketches. Results shown in Fig.10 demonstrate that the baseline model frequently fails to produce realistic textures. In contrast, our results are more realistic with fine textures.

Our model can achieve fine-grained control on generated images, whereas baseline model cannot. When we modify the lines of face parts or face shapes of input freehand sketches, the corresponding parts of images generated by our model change simultaneously but other areas remain unchanged. In comparison, modifying strokes of sketches influences not only the content in corresponding areas of images generated by baseline model but also the content in other areas. Fig.11 and Fig.12 show several face images generated by our model and baseline model when changing lines of mouth and nose separately in sketches. The results shown in row 1, 2 of Fig.11 demonstrate that the images generated by our model change obviously in

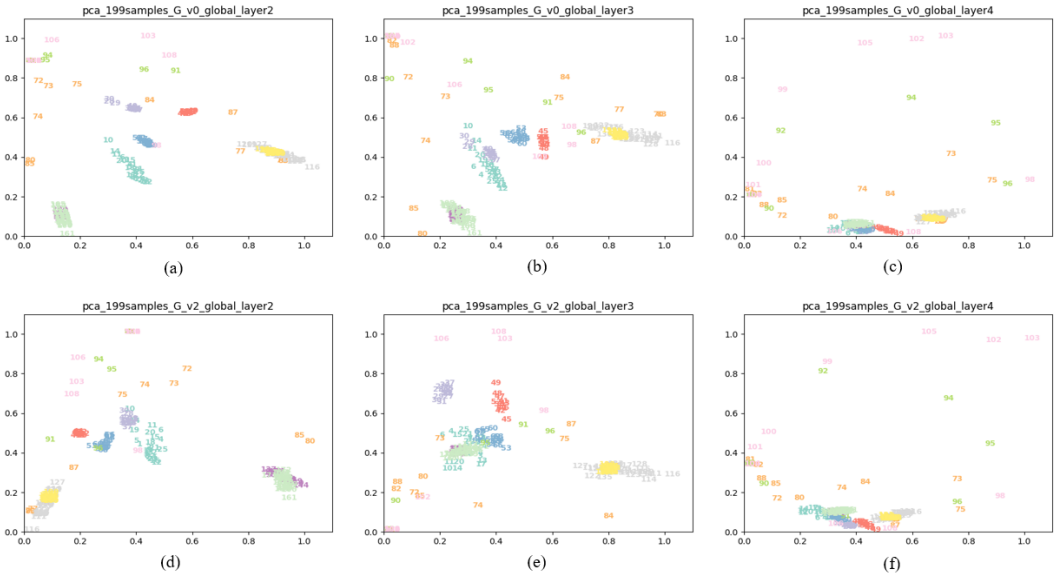


Figure 6. Comparative PCA visualization results of $\boldsymbol{v}_2 \sim \boldsymbol{v}_4$ and $\boldsymbol{v}_2^{'} \sim \boldsymbol{v}_4^{'}$

Figure 7. The average face of all the training data.

Figure 8. Comparison between images generated before and after data augmentation. The quality of generated images after data augmentation is better than that before data augmentation when the inputs deviate from standard position.

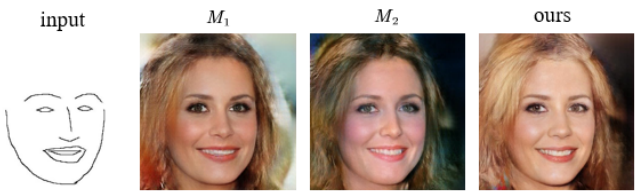


Figure 9. Comparison on amount of instance normalization layers. We refer to original pix2pixHD model as M_1, refer to the model of which generator gets rid of first 5 IN layers as M_2.

mouth shape and preserve structure conformance with input sketches when modifying the lines of mouth in sketches. But the results generated by baseline model do not have an obvious change in mouth shape. The results shown in Fig.12 illustrate that the images generated by our model remain unchanged in other areas especially in eyes when altering the shapes of nose in

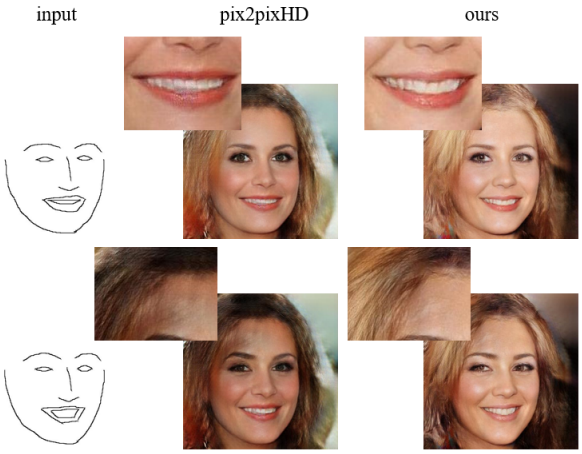


Figure 10. Magnified details of images generated by our model and baseline model. The top row shows that the textures of teeth generated by pix2pixHD baseline are blurry, while our results have clear textures of teeth. The bottom row shows that some chaotic noises often emerge in the forehead of images generated by pix2pixHD baseline, but our results do not have this issue.

sketches, but the images generated by baseline model change obviously in eyes direction. And as shown in row 3 of Fig.11, the generation results of our model do not change except the mouth when modifying the mouth shape in sketches, while the generation quality of baseline model is degraded vastly especially in the eye area.

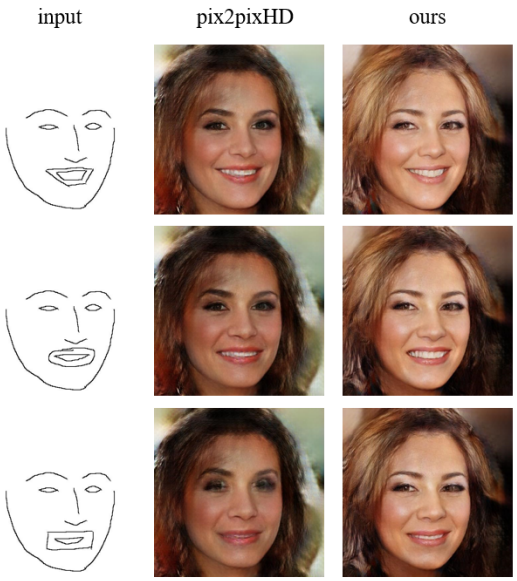


Figure 11. Comparison between our model and baseline model tested with mouth-altered sketches.

5. Conclusion

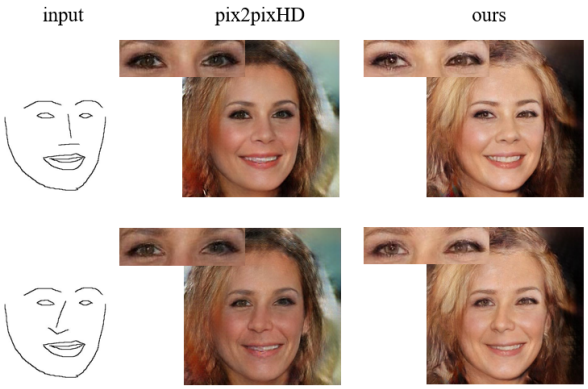


Figure 12. Comparison between our model and baseline model tested with nose-altered sketches.

References

[1] Chen Qifeng and Koltun Vladlen. Photographic image synthesis with cascaded refinement networks. In The IEEE International Conference on Computer Vision (ICCV), Oct 2017. 2

[2] Citation. Hotelling H. . Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, 24(6):417–441, 1933. 1, 3

[3] H. Emami, M. M. Aliabadi, M. Dong, and R. B. Chinnam. Spa-gan: Spatial attention gan for image-to-image translation, 2020. 2

[4] Goodfellow Ian, Pouget-Abadie Jean, Mirza Mehdi, Xu Bing, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Advances in Neural Information Processing Systems(NIPS), pages 2672–2680. 2014. 1

[5] Hao Dong, Simiao Yu, Chao Wu, and Yike Guo. Semantic image synthesis via adversarial learning. In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pages 5707–5715, 2017. 2

[6] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In ECCV, 2018. 2

[7] Isola Phillip, Zhu Jun-Yan, Zhou Tinghui, and Efros Alexei A. Image-to-image translation with conditional adversarial networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017. 2, 3

[8] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In International Conference on Computer Vision(ICCV), pages 2242–2251, 2017. 1, 2

[9] Lee, Cheng-Han, Liu, Ziwei, Wu, Lingyun, and Luo, Ping. Maskgan: Towards diverse and interactive facial image manipulation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020. 2, 5

[10] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. CoRR, abs/1607.06450, 2016. 1, 2

[11] Li, Yuhang, Chen, Xuejin, Wu, Feng, and Zha, Zheng-Jun. Linestofacephoto: Face photo generation from lines with conditional self-attention generative adversarial networks. In Proceedings of the 27th ACM International Conference on Multimedia, pages 2323–2331, 2019. 5

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International Conference on Machine Learning(ICML), volume 37, pages 448–456, 2015. 1, 2

[13] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In Conference on Computer Vision and Pattern Recognition(CVPR), pages 2337–2346, 2019. 1, 2

[14] Takuhiro Kaneko, Kaoru Hiramatsu, and Kunio Kashino. Generative attribute controller with conditional filtered generative adversarial networks. In Conference on Computer Vision and Pattern Recognition(CVPR), pages 7006–7015, 2017. 2

[15] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. CoRR, abs/1710.10196, 2017. 5

[16] Ulyanov Dmitry, Vedaldi Andrea, and Lempitsky Victor. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017. 1, 2

[17] Wang Ting-Chun, Liu Ming-Yu, Zhu Jun-Yan, Tao Andrew, Kautz Jan, and Catanzaro Bryan. High-resolution image synthesis and semantic manipulation with conditional gans. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 1, 2

[18] Yuxin Wu and Kaiming He. Group normalization. In European Conference of Computer Vision(ECCV), volume 11217, pages 3–19, 2018. 1, 2

[19] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In Advances in Neural Information Processing Systems, 2017. 2