Generative Attribute Controller with Conditional Filtered Generative Adversarial Networks - Supplementary Material

Takuhiro Kaneko Kaoru Hiramatsu Kunio Kashino
NTT Communication Science Laboratories, NTT Corporation
{kaneko.takuhiro, hiramatsu.kaoru, kashino.kunio}@lab.ntt.co.jp

A. Detailed Experimental Setup

In this section, we describe the network architectures and training procedures for each dataset. In the following tables, *FC.* and *conv.* indicate fully connected and convolution, respectively. The symbols $\downarrow$ and $\uparrow$ represent downsampling and upsampling, respectively. To downscale and upscale, we respectively used convolutions and backward convolutions (i.e., fractionally strided convolutions) with stride 2. The terms *BNorm*, *ReLU*, and *LReLU* indicate batch normalization [1], rectified linear unit activation [5], and leaky rectified linear unit activation [4, 8], respectively. To make a discriminator D and an auxiliary network Q conditioned on attribute y, we extended and reshaped y to fit the size of the layer and concatenated it to the networks. This is indicated as $+$ *cond.* in the tables. We trained the models using the Adam optimizer [2], with a minibatch of size 128. We set the learning rate as 2e-4 for D/Q, 1e-3 for the generator network G, and 1e-3 for the classifier/encoder C/E. The momentum term β_1 was set to 0.5. The details for each dataset are described below.

A.1. MNIST

The network architectures are shown in Table 1. The D and Q share most parts of the network. For this task, we used a 100-dimensional attribute-independent variable z_i. For attribute-dependent variable z'_a, we used three types: a ten-categorical RB, two-dimensional SB-I, and two-dimensional SB-II. We compared them in the experiments. In the pre-experiments, we found that the D tended to overfit to the training data, so we added 0.001 weight decay to the discriminator objective function.

A.2. CUB

The network architectures are shown in Table 2. The D and Q share most parts of the network. For this task, we used a 128-dimensional z_i. For z'_a, we used the combination of a five-categorical RB and one-dimensional SB-I. In the pre-experiments, we found that the D tended to overfit to the training data, so we added 0.004 weight decay to the discriminator objective function.

Generator G
Input: $z_i \in \mathbb{R}^{100}$, $z'_a (= f_y(z_a))$
1024 FC., BNorm, ReLU
$7 \cdot 7 \cdot 128$ FC., BNorm, ReLU
4×4 64 conv. $\uparrow$, BNorm, ReLU
4×4 1 conv. $\uparrow$, sigmoid

Discriminator D **/ Auxiliary Network** Q
Input: 28×28 gray image $+$ cond.
4×4 64 conv. $\downarrow$, LReLU $+$ cond.
4×4 128 conv. $\downarrow$, BNorm, LReLU $+$ cond.
1024 FC., BNorm, LReLU $+$ cond.
FC. output for D 128 FC.-BNorm-LReLU-FC. output for Q

Table 1. Network architectures used for MNIST dataset

Generator G
Input: $z_i \in \mathbb{R}^{128}$, $z'_a (= f_y(z_a))$
$4 \cdot 4 \cdot 512$ FC., BNorm, ReLU
4×4 256 conv. $\uparrow$, BNorm, ReLU
4×4 128 conv. $\uparrow$, BNorm, ReLU
4×4 64 conv. $\uparrow$, BNorm, ReLU
4×4 3 conv. $\uparrow$, tanh

Discriminator D **/ Auxiliary Network** Q
Input: 64×64 color image $+$ cond.
4×4 64 conv. $\downarrow$, LReLU $+$ cond.
4×4 128 conv. $\downarrow$, BNorm, LReLU $+$ cond.
4×4 256 conv. $\downarrow$, BNorm, LReLU $+$ cond.
4×4 512 conv. $\downarrow$, BNorm, LReLU $+$ cond.
FC. output for D 128 FC.-BNorm-LReLU-FC. output for Q

Table 2. Network architectures used for CUB dataset

Generator G
Input: $z_i \in \mathbb{R}^{128}, z'_a (= f_y(z_a))$
$4 \cdot 4 \cdot 512$ FC., BNorm, ReLU
4×4 256 conv. $\uparrow$, BNorm, ReLU
4×4 128 conv. $\uparrow$, BNorm, ReLU
4×4 64 conv. $\uparrow$, BNorm, ReLU
4×4 3 conv. $\uparrow$, tanh

Discriminator D / Auxiliary Network Q
Input: 64×64 color image + cond.
4×4 64 conv. $\downarrow$, LReLU + cond.
4×4 128 conv. $\downarrow$, BNorm, LReLU + cond.
4×4 256 conv. $\downarrow$, BNorm, LReLU + cond.
4×4 512 conv. $\downarrow$, BNorm, LReLU + cond.
FC. output for D 128 FC.-BNorm-LReLU-FC. output for Q

Classifier C / Encoder E
Input: 64×64 color image
4×4 64 conv. $\downarrow$, LReLU
4×4 128 conv. $\downarrow$, BNorm, LReLU
4×4 256 conv. $\downarrow$, BNorm, LReLU
4×4 512 conv. $\downarrow$, BNorm, LReLU
FC. output for C FC. output for E

Table 3. Network architectures used for CelebA dataset

A.3. CelebA

The network architectures are shown in Table 3. The D and Q share most parts of the network. The C and E also share most parts of the network. For this task, we used a 128-dimensional z_i. For z'_a, we used three types: a ten-categorical RB, three-dimensional SB-I, and three-dimensional SB-II. We compared them in the experiments.

B. Extended Results

In the main paper, we reported only the key results due to space limitations. This section provides more results and comparisons to clarify the characteristics of our CFGAN.

B.1. Attribute-based Image Generation

B.1.1 Control between Two Digits

As described in the main paper, we used the subsets of the MNSIT dataset to clarify the basic characteristics of the CF-GAN. We selected two types of digits and regarded one as an attribute ($y = 1$) and the other as a non-attribute ($y = 0$). In the main paper, we showed the results in which "4" and "9" were regarded as the attribute and non-attribute, respectively. We now give the other cases. Figure 3 shows the results in which "1" and "7" were regarded as the attribute and non-attribute, respectively, Figure 4 shows the results in which "3" and "8" were regarded as the attribute and non-attribute, respectively, and Figure 5 shows the results in which "5" and "6" were regarded as the attribute and non-attribute, respectively. For each case, we implemented three controllers (a ten-categorical RB, two-dimensional SB-I, and two-dimensional SB-II) and compared them.

These results indicate that the CFGAN could represent large variations of an attribute regardless of the digits and that the variations could be controlled categorically with the RB and continuously with the SB-I or the SB-II. In particular, the SB-I enabled attributes to be controlled continuously around the non-attribute state, and the SB-II enabled them to be controlled continuously from non-attribute to attribute states. In both the CGAN and CFGAN + RB, the identities (e.g., line width and rough shape) were retained. The difference between them is that the CGAN changes digits deterministically but the CFGAN enables a user to select the best one using the controllers.

B.1.2 Red-Bird and Yellow-Bird Generators

In the main paper, we discussed "blue"-bird generation. We now discuss the other colored bird generations. Figures 6 and 7 show "red"-bird and "yellow"-bird generations, respectively. For "red"-bird generation, we selected the attributes with "red" in the name (e.g., red wing, red tail, red eye, red leg, and red crown) from 312 attributes, summarized them all as "red" birds, and used them as the supervised data, i.e., regarded "red" as the attribute ($y = 1$) and "not red" as the non-attribute ($y = 0$). We used the CF-GAN with the combination of a five-categorical RB and one-dimensional SB-I. For "yellow"-bird generation, we conducted the experiments using a similar setting as that of "red"-bird generation.

Similar to digit generation, the CFGAN enables a user to control large variations of "red" birds and "yellow" birds. For our experiments, in particular, we designed a controller with a combination of the RB and SB-I, which enabled images to be controlled categorically and continuously at the same time. The SB-I controller enables a user to control the degree of color, and the RB controller enables a user to select the place to be colored. Note that, in training the models, we only use the binary value indicating the presence or absence of an attribute (e.g., red or not) as the supervised data, and do not use the degree of color or position of the colored place as the supervised data.

B.1.3 Face-Attribute Generator

To clarify the difference in controllers for various attributes, we implemented three controllers (a ten-categorical RB, three-dimensional SB-I, and three-dimensional SB-II) for six attributes (bangs, eyeglasses, heavy makeup, male, smiling, and young). The results are shown in Figures 8-13.

In the GAN, it is difficult to control whether a generated image includes the attribute because its latent variables are highly entangled. The CGAN can control the attribute by switching the value of y, but it cannot control variations of the attribute because y represents only the presence or absence of the attribute. In contrast, the CFGAN enables variations of the attribute to be controlled: the RB enables them to be controlled categorically, the SB-I enables them to be controlled continuously around the non-attribute state, and SB-II enables them to be controlled continuously from non-attribute to attribute state. Only the CFGAN + three-dimensional SB-II for glasses did not work well. The reason is considered that, in this case, it was difficult to define the strength of the non-attribute state (i.e., the strength of non-glasses state), so the continuous representation between the attribute and non-attribute states did not fit well. This result indicates that it would be better to select the types of controllers according to the attribute property.

B.1.4 Comparison of Different Categorical RBs

To evaluate the relationship between the number of categories and controllability of RB, we compared the RBs in which the number of categories were different. Figure 14 shows example images generated using them. We selected “glasses” as an attribute. When the number of categories was small (e.g., (a) and (b)), the different types of glasses were represented in the same category. In contrast, as the number of categories increased, the slightly different glasses became separated. Generally speaking, there is a trade-off between abstract and detailed description, and it is difficult to determine the best one. Therefore, it is recommended to select the number of categories depending on the application.

B.1.5 Comparison of CFGAN and InfoCGAN

As discussed in Sec. 4.2 of the main paper, the fundamental difference between the CFGAN and InfoCGAN is whether the controllable variable (i.e., z'_a in the CFGAN and z_c in the InfoCGAN) is conditioned on y.

To further clarify this difference, we conducted a quantitative analysis to measure how the generated variations are specific to the attribute. We first generated two images x^1 and x^2 using the same z_i, same $y = 1$, and randomly sampled the controllable variable (i.e., z'_a in the CFGAN and z_c in the InfoCGAN). We then cropped the attribute-specific and unspecific areas, as shown in Figure 1, and calculated SSIM [7] between pair images for each area. Finally, we calculated their ratio: $VS(x^1, x^2) = \text{SSIM}(x^1_{\text{attr}}, x^2_{\text{attr}})/\text{SSIM}(x^1_{\text{other}}, x^2_{\text{other}})$. A smaller score indicates that variations are specific to the attribute. We calculated the scores for 100,000 randomly generated pair images and took the average. We call this score the *variation specialization (VS)* score. For comparison, we implemented five models (InfoCGAN with a ten-categorical RB, InfoCGAN with a three-dimensional SB, CFGAN with a ten-categorical RB, CFGAN with a three-dimensional SB-I, and CFGAN with a three-dimensional SB-II) for two attributes (glasses and bangs). Table 4 lists the results and indicates that the variations in the CFGANs are more specific to the attribute than those in the InfoCGANs.

We also show the extended qualitative results in Figure 2. We regarded “4” as the attribute ($y = 0$) and “9” as the non-attribute ($y = 1$). We implemented the following three models: InfoCGAN with a two-dimensional SB, CFGAN with a two-dimensional SB-I, and CFGAN with a two-dimensional SB-II. In the InfoCGAN, z_c was not conditioned on y; therefore, attribute-independent features (i.e., features common between “4” and “9”) were obtained. For

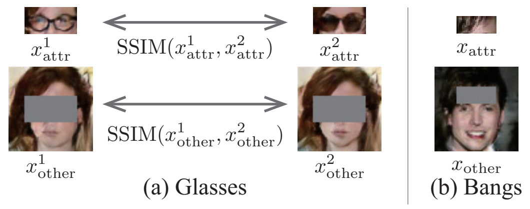


Figure 1. Definition of attribute-specific and unspecific areas

	Glasses	Bangs
InfoCGAN (RB)	1.14	1.07
InfoCGAN (SB)	1.04	0.95
CFGAN (RB)	0.65	0.48
CFGAN (SB-I)	0.52	0.43
CFGAN (SB-II)	0.92	0.82

Table 4. Variation specialization scores

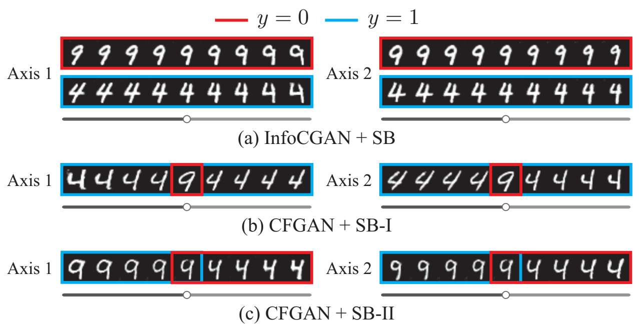


Figure 2. Sample images generated using InfoCGAN and CFGANs. In (a), sample images were generated from same z_i and varying z_c using InfoCGAN, while in (b)–(c), sample images were generated from same z_i and varying z'_a using CFGANs.

example, in Figure 2 (a), "rotation" and "thickness" could be controlled independently from the attribute. In the CFGAN, z'_a was conditioned on y using the filtering architecture; therefore, attribute-dependent features were obtained. For example, in Figure 2 (b), the attribute state was continuously controlled around the non-attribute state, while in Figure 2 (c), the non-attribute and attribute states were continuously controlled.

B.2. Attribute-based Image Editing

We applied the same face-attribute generator described in Sec. B.1.3 to the image-editing task. For each attribute, we selected one controller based on the results in Sec. B.1.3. Figures 15-20 show the results. To achieve image editing, we conducted the procedure described in Figure 5 of the main paper, i.e., (1) load an input image, (2) reconstruct with encoder neural network, (3) reconstruct with gradient descent, (4) modify latent variables, and (5) apply post-processing.

As discussed in a previous study [3], it is challenging for a deep generative model to modify attributes while retaining the identity of an input image because the detailed texture could be lost in the reconstruction process. To alleviate this problem, we developed a post-processing technique and it works well as shown in Figures 15-20. Note that this technique does not depend on the CFGAN; therefore, it can be used for general generative models. Moreover, previous deep generative models such as CVAE [9] and VAE/GAN + visual attribute vector [3] controlled each attribute one-dimensionally, similar to the CGAN, but the CFGAN enabled an attribute to be controlled multi-dimensionally. This enables a user to edit attributes of an image more expressively.

B.3. Attribute Transfer

In this section, we provide the extended results of attribute transfer. Figure 21 shows the results in which the attributes (bangs) of ten people were transferred to ten other people. We used the CFGAN with a three-dimensional SB-I for this task. The column shows the sample images generated from the same z'_a for the different z_is. These results indicate that hair style (e.g., left parted hair, bob cut hair) can be transferred regardless of gender, age, and pose. The row shows the sample images generated from the same z_i for the different z'_as. These results indicate that identity is retained regardless of transferred hair style.

In fact, previous studies [3, 6] have also attempted to transfer attributes between individuals using deep generative models. However, they required multiple samples that contain the same attribute because their latent variables are not disentangled into attribute-dependent and attribute-independent, and they need to average them to know the "secret" of the attribute. In the CFGAN, however, the attribute-dependent and attribute-independent variables are disentangled in the latent space, so we can do "one-shot" transfer by exchanging only z'_as.

Attributes	Controller type
Bangs	three-dimensional SB-I
Glasses	ten-categorical RB
Makeup	three-dimensional SB-I
Male	three-dimensional SB-II
Smiling	three-dimensional SB-II
Young	three-dimensional SB-II

Table 5. Sets of attributes and controller types in CFGAN with multiple attribute controllers, on which interface shown in Figure 23 was based.

B.4. Attribute-based Image Retrieval

Figure 22 shows the extended results of attribute-based image retrieval. We used the CFGAN with a three-dimensional SB-I for this task. For comparison, we measured the distance in the low-level image space, i.e., x, attribute-label space, i.e., y, attribute-independent latent variable space, i.e., z_i, and attribute-dependent latent variable space, i.e., z'_a. When images were retrieved on the basis of x or z_i, images that were globally similar to the query (e.g., background and pose) were retrieved. In contrast, when images were retrieved on the basis of y or z'_a, images that had the same attribute as the query (i.e., "with glasses" in this case) were retrieved. In particular, when images were retrieved on the basis of y, the type of attribute (e.g., thin glasses, sunglasses, thick glasses) was not consistent; however, when images were retrieved on the basis of z'_a, the type of attribute was similar to the query. These results indicate that the learned latent variable, i.e., z'_a has enough expressive power to represent variations of "glasses". Note that we did not use the type of attribute as the supervised data; rather, we only used binary value indicating the presence or absence of an attribute as the supervised data. Based on the results, we plan to use the CFGAN as a latent variation-discovery tool for high-dimensional data.

C. Interface for GAC

We developed an interface for a GAC with the CFGAN to show its effectiveness. We used the CFGAN with multiple attribute controllers for this task. The sets of attributes and controller types are listed in Table 5. We conducted the experiment with a laptop PC. Once the latent variables were estimated, we used only a feedforward neural network and simple post-processing. This enabled us to edit an image in real time. Figure 23 shows image-editing examples using this interface.

References

[1] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 1

[2] D. P. Kingma and M. Welling. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

[3] A. B. L. Larsen, S. K. Sønderby, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016. 4

[4] A. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 1

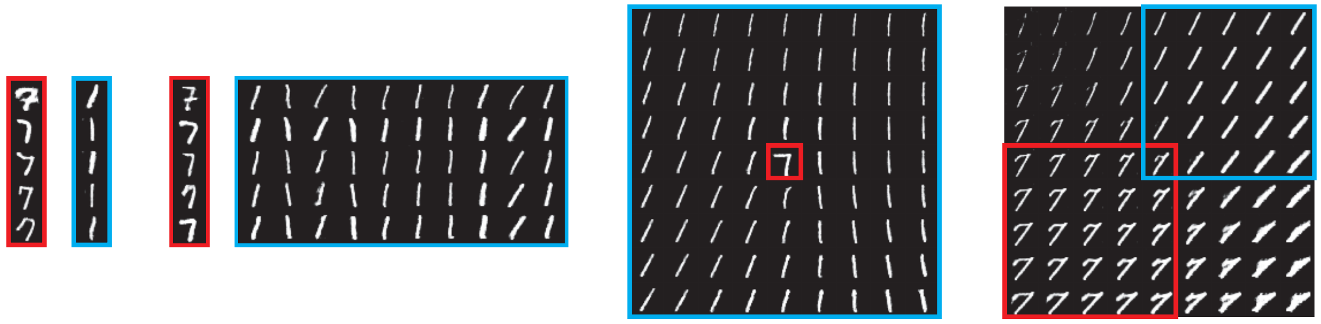
[5] V. Nair and G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *ICML*, 2010. 1

[6] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 4

[7] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on IP*, 13(4):600–612, 2004. 3

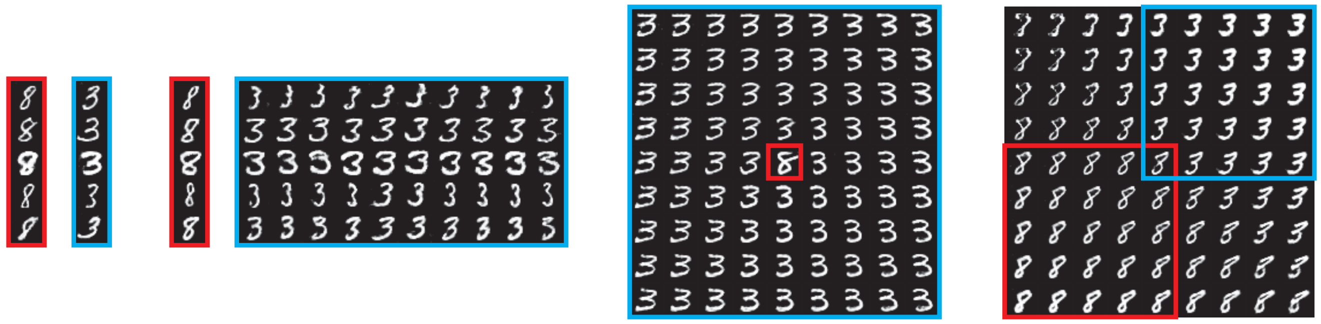
[8] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. In *ICML Workshop*, 2015. 1

[9] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2Image: Conditional image generation from visual attributes. In *ECCV*, 2016. 4



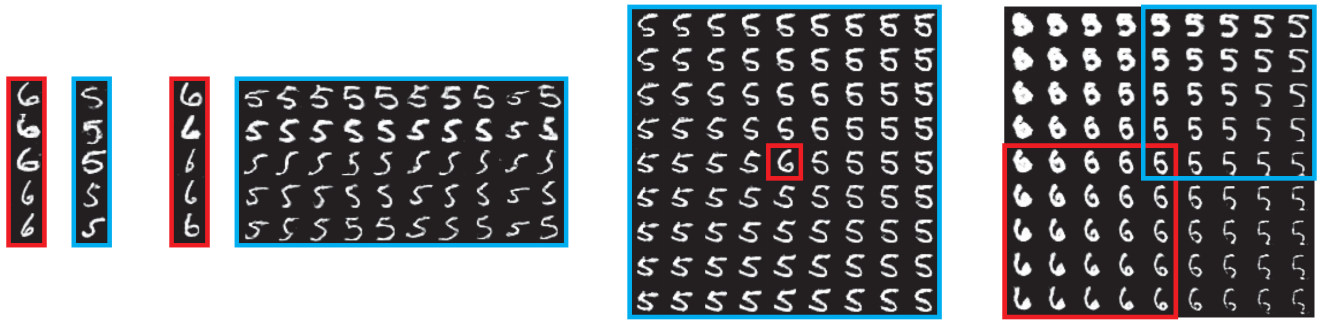
(a) CGAN (b) CFGAN + RB (c) CFGAN + SB-I (d) CFGAN + SB-II

Figure 3. Example results of control between 1 and 7. Digits in red boxes do not contain attributes ($y = 0$), while digits in blue boxes contain attributes ($y = 1$). In (a) and (b), row shows sample images generated from same z and z_i, respectively. In right figure of (b), column contains five samples from same category in z'_a. In (c) and (d), z'_a changed continuously along two orthogonal axes.



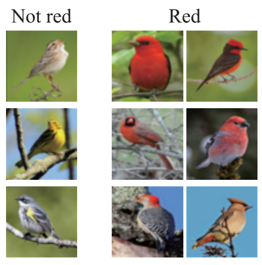
(a) CGAN (b) CFGAN + RB (c) CFGAN + SB-I (d) CFGAN + SB-II

Figure 4. Example results of control between 3 and 8. View of figure is same as that in Figure 3.

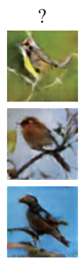


(a) CGAN (b) CFGAN + RB (c) CFGAN + SB-I (d) CFGAN + SB-II

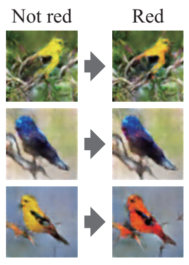
Figure 5. Example results of control between 5 and 6. View of figure is same as that in Figure 3.



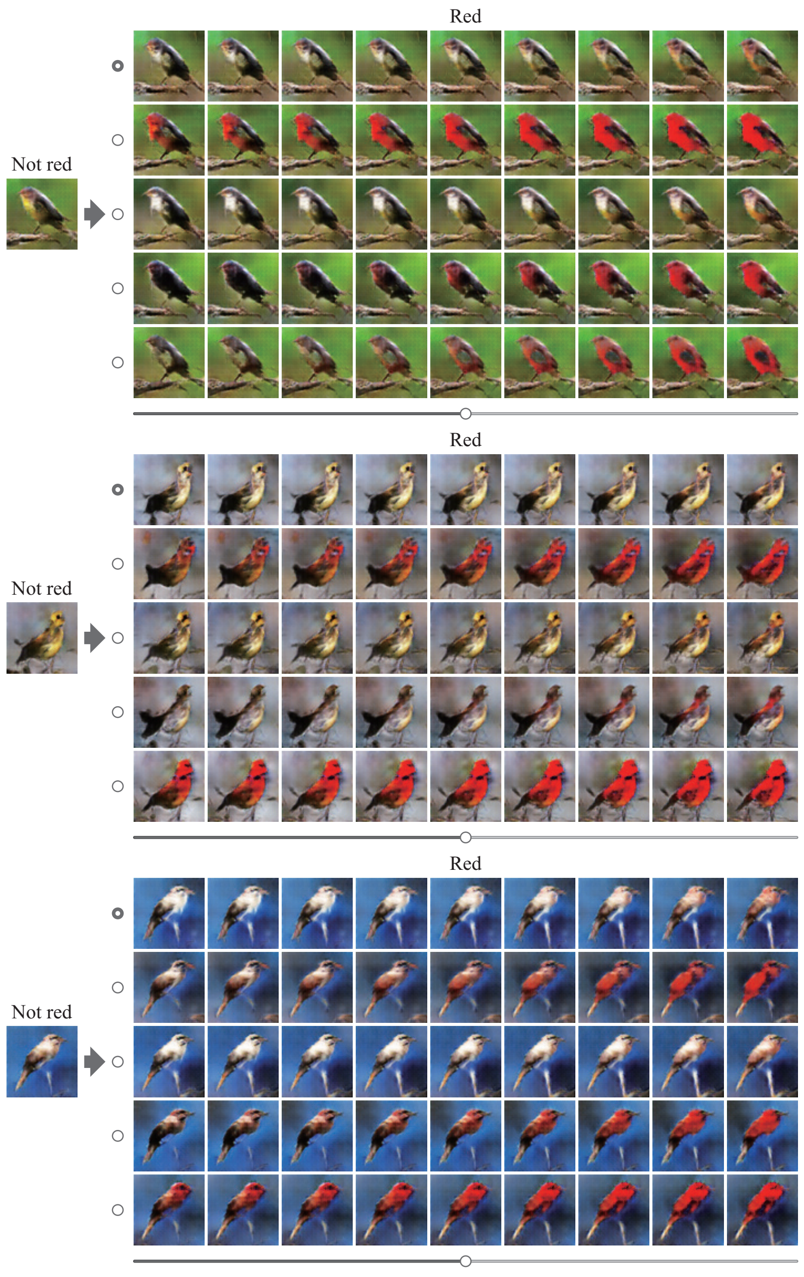
(a) Original



(b) GAN



(c) CGAN



(d) CFGAN

Figure 6. Example results of red-bird generator. In (d), row shows samples from same category of RB while varying value of SB-I continuously from left to right.

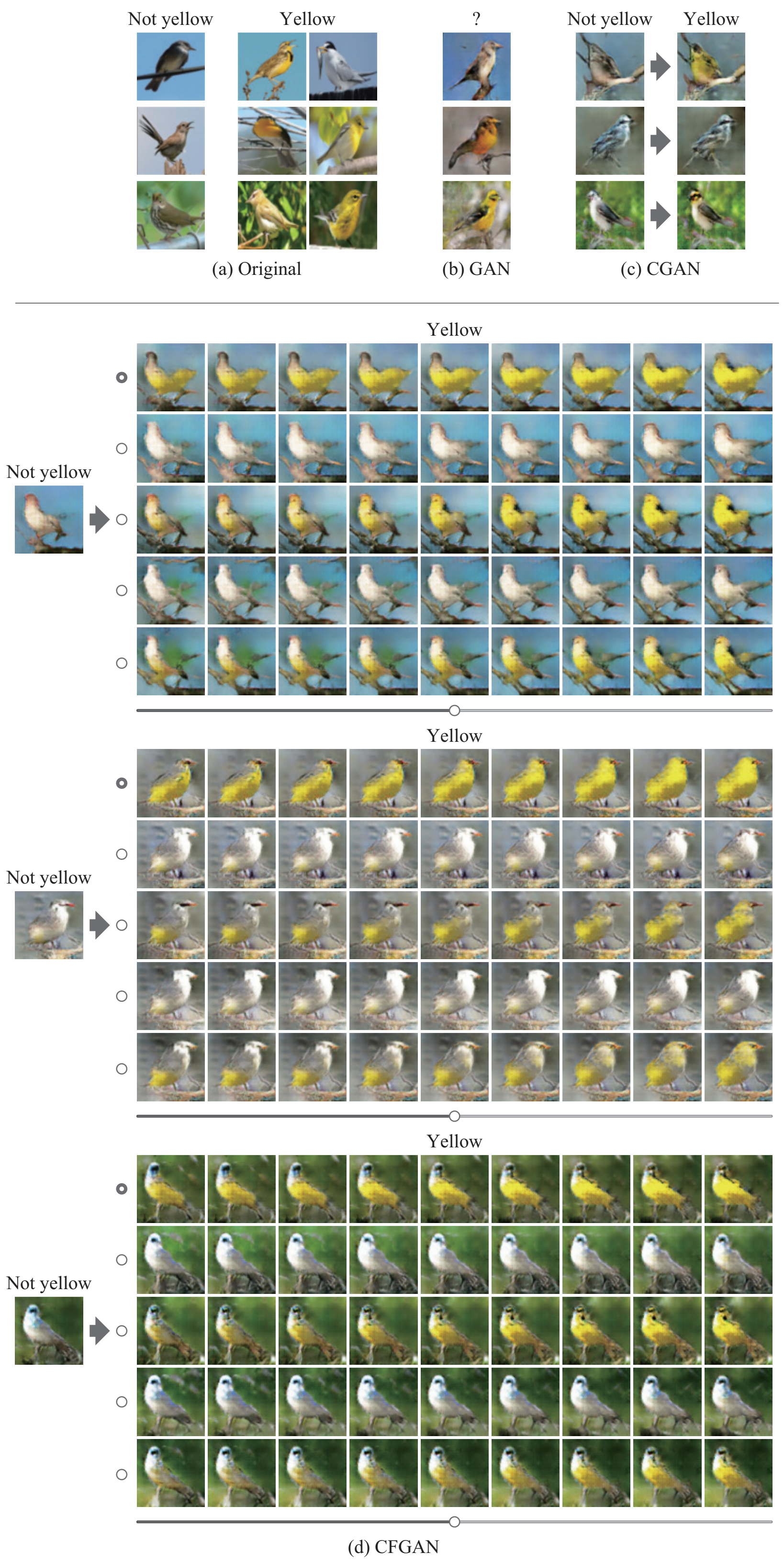
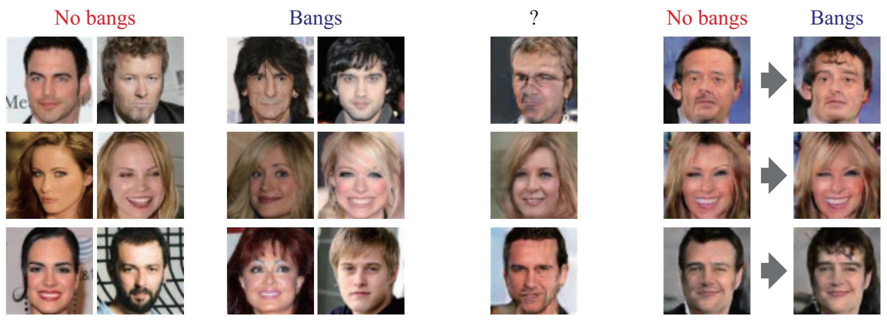
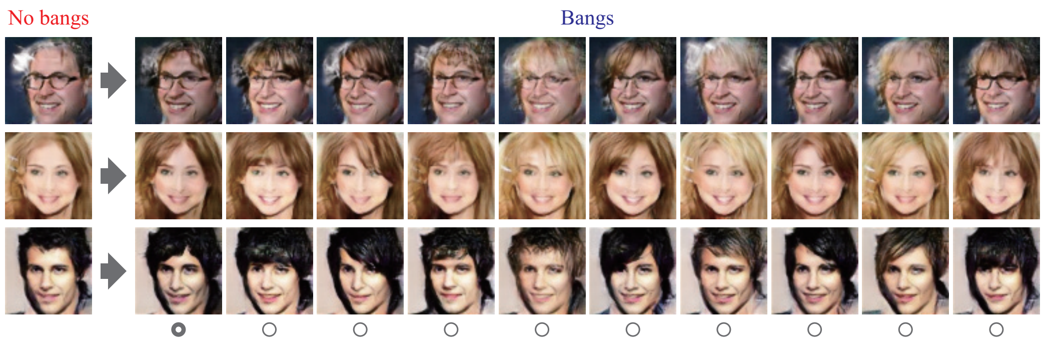


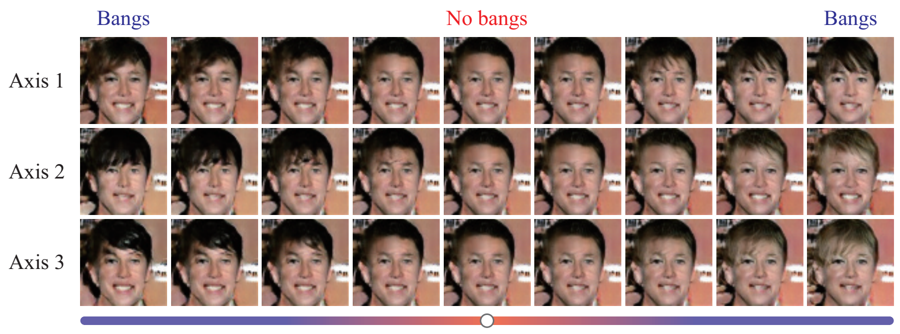
Figure 7. Example results of yellow-bird generator. View of figure is same as that in Figure 6.



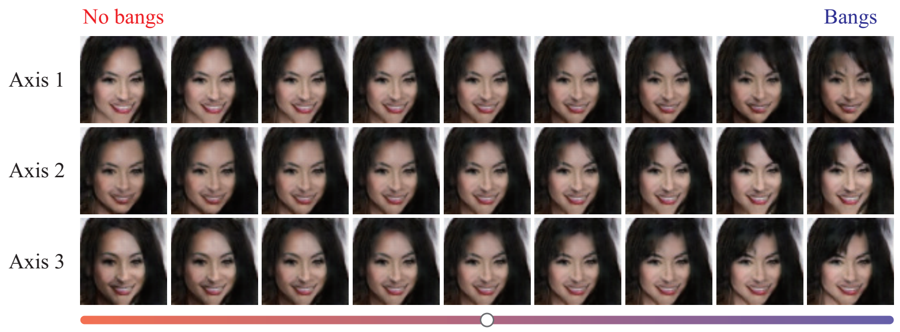
(a) Original (b) GAN (c) CGAN



(d-1) CFGAN + ten-categorical RB



(d-2) CFGAN + three-dimensional SB-I



(d-3) CFGAN + three-dimensional SB-II

Figure 8. Example results of bangs generation. In (c), row shows sample images from same z, and column shows images generated for $y = 0$ or $y = 1$. In (d-1), row shows sample images from same z_i, and column shows images generated for different z'_a. In (d-2) and (d-3), sample images are generated from same z_i. Row shows sample images where value of SB-I was varied continuously from left to right for each axis.

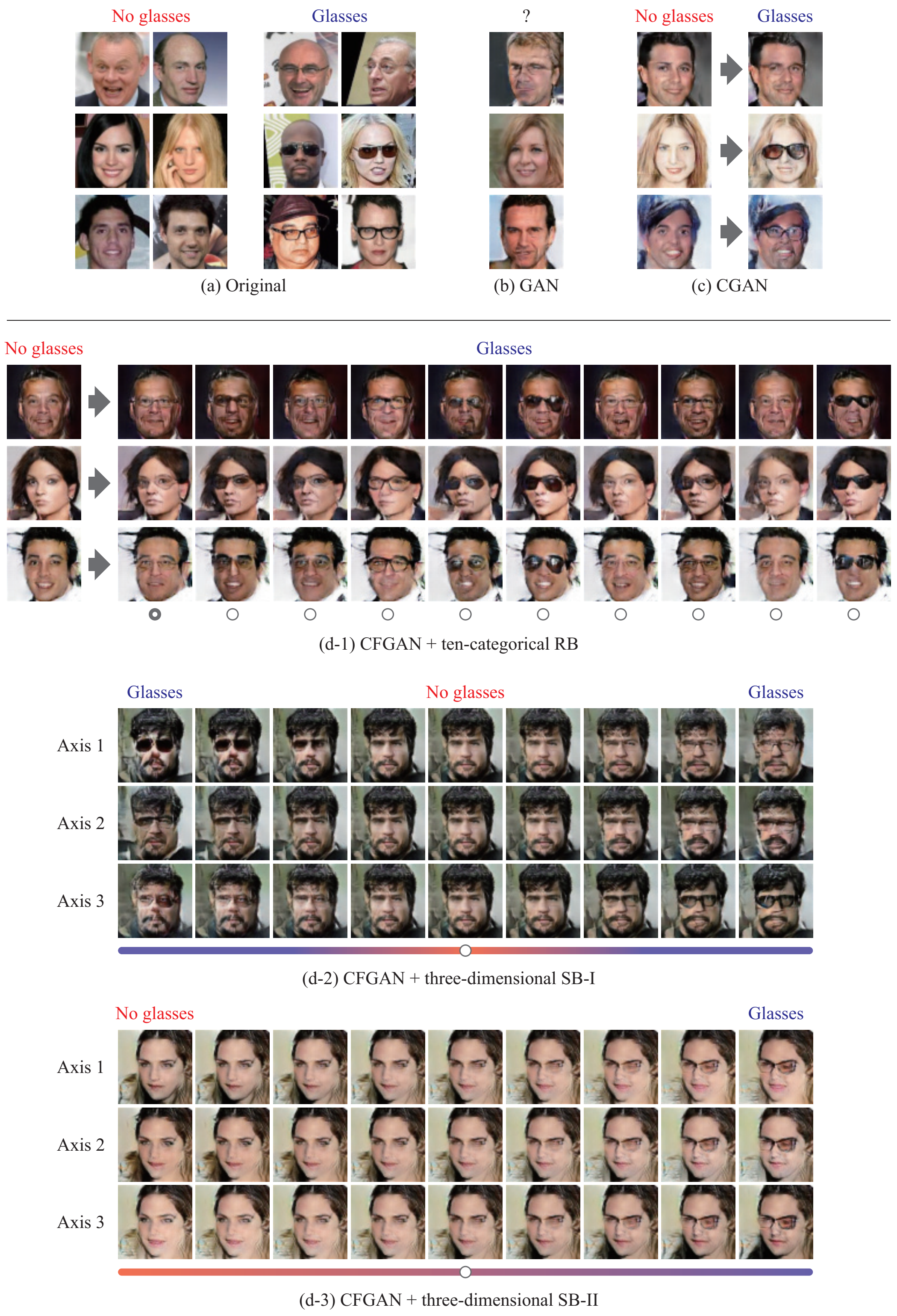
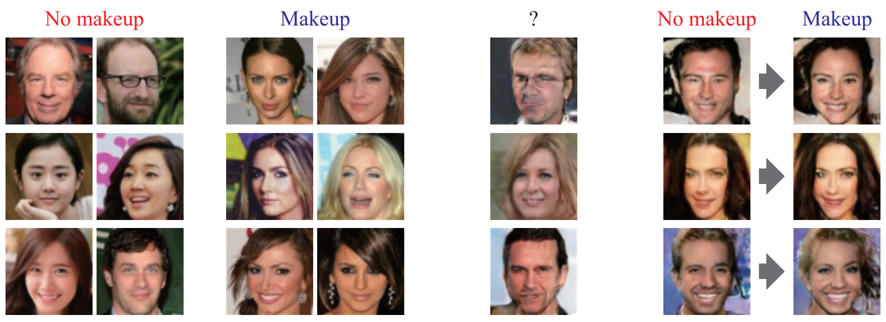
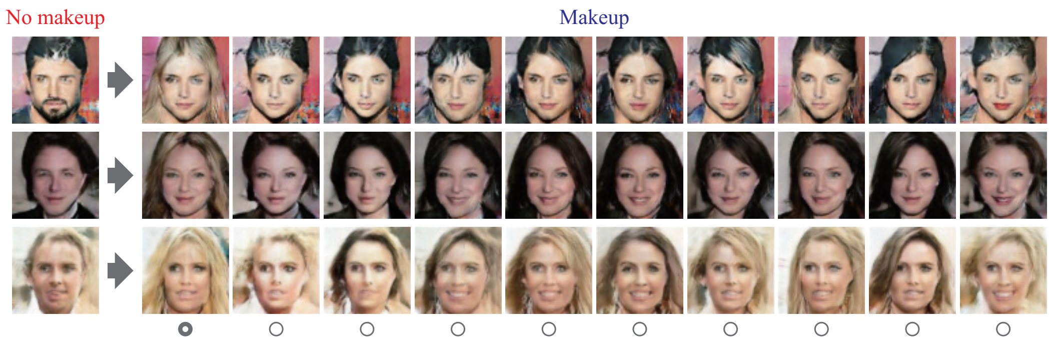


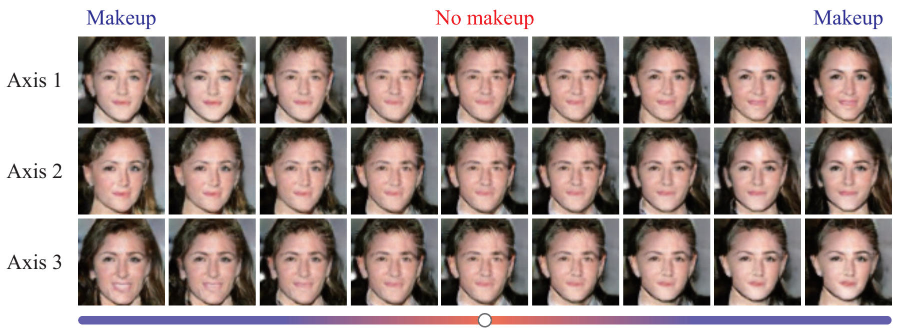
Figure 9. Example results of glasses generation. View of figure is same as that in Figure 8.



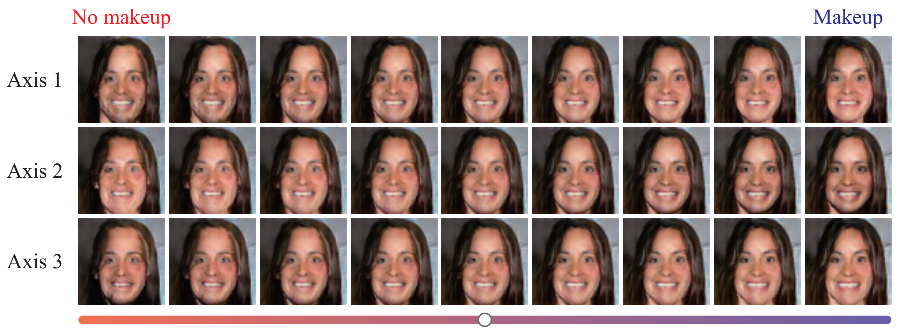
(a) Original (b) GAN (c) CGAN



(d-1) CFGAN + ten-categorical RB

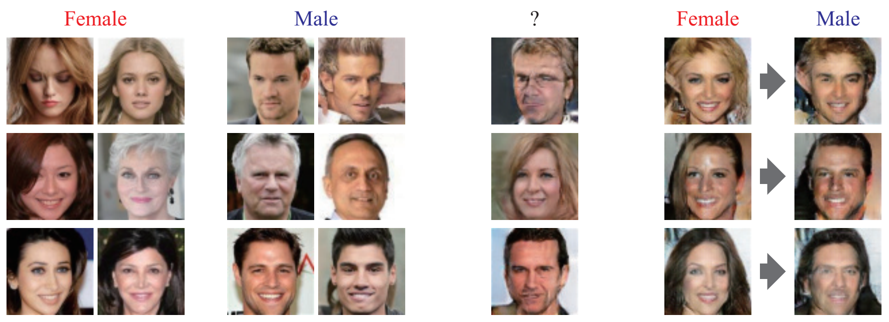


(d-2) CFGAN + three-dimensional SB-I

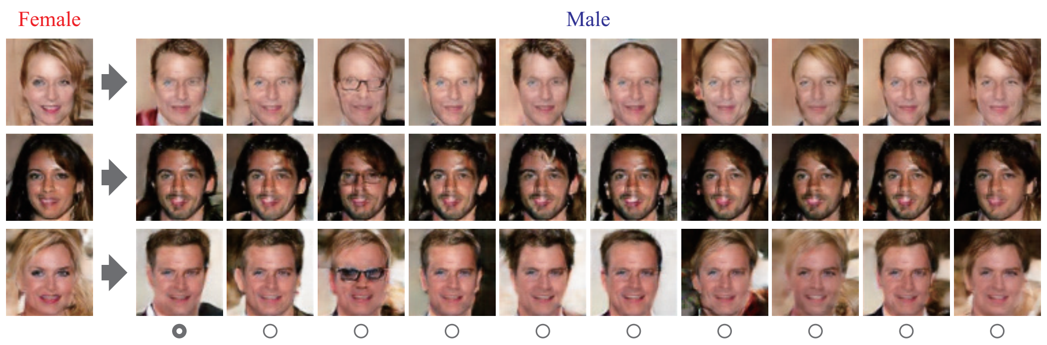


(d-3) CFGAN + three-dimensional SB-II

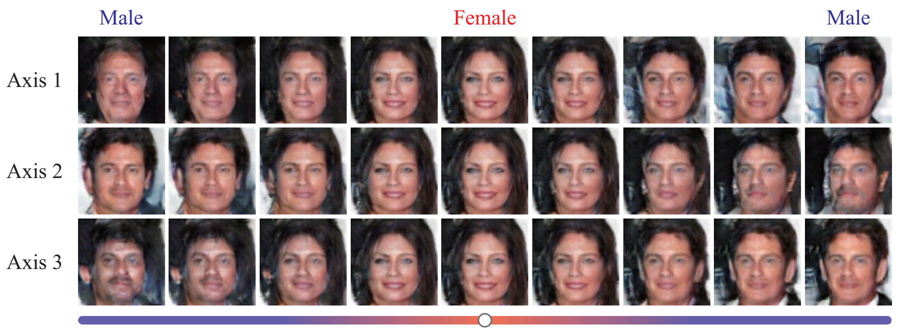
Figure 10. Example results of makeup generation. View of figure is same as that in Figure 8.



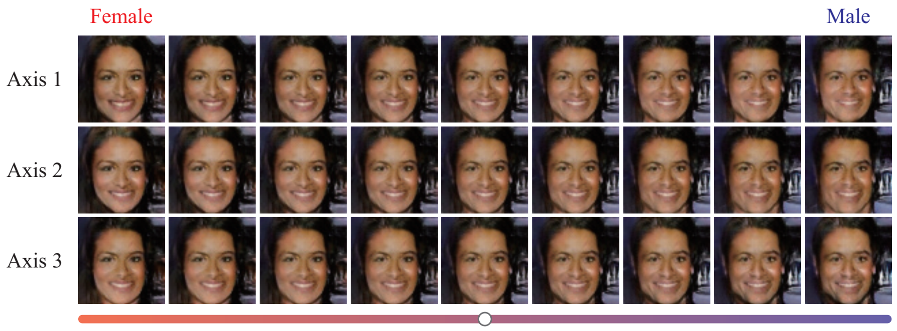
(a) Original (b) GAN (c) CGAN



(d-1) CFGAN + ten-categorical RB



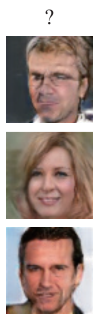
(d-2) CFGAN + three-dimensional SB-I



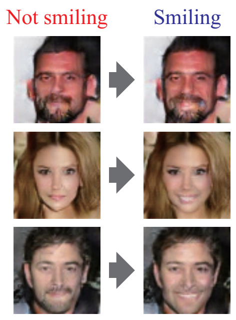
(d-3) CFGAN + three-dimensional SB-II

Figure 11. Example results of male generation. View of figure is same as that in Figure 8.

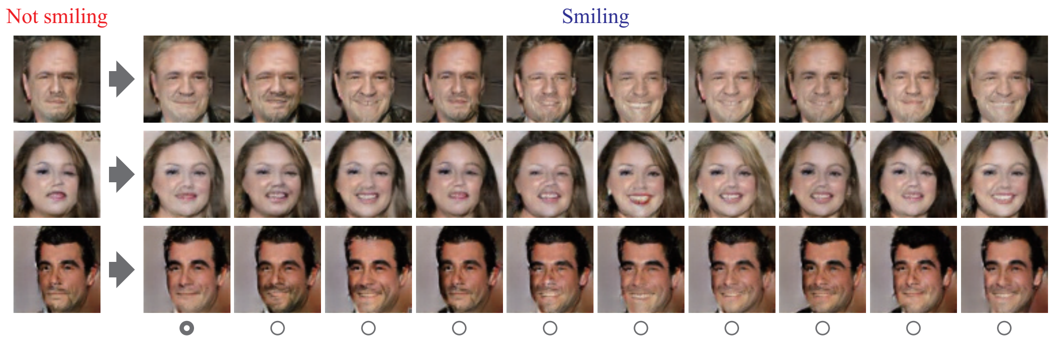
(a) Original



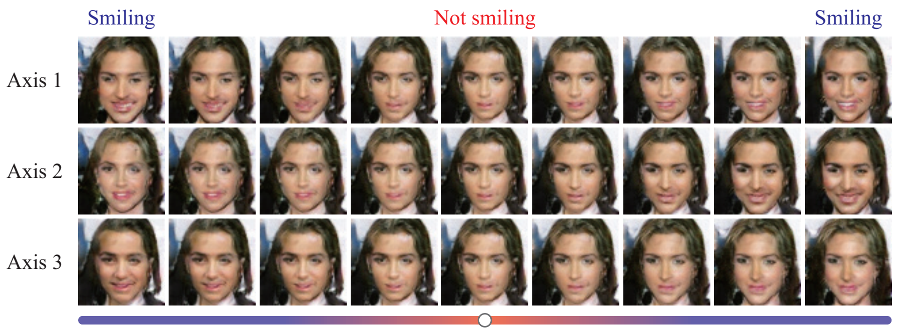
(b) GAN



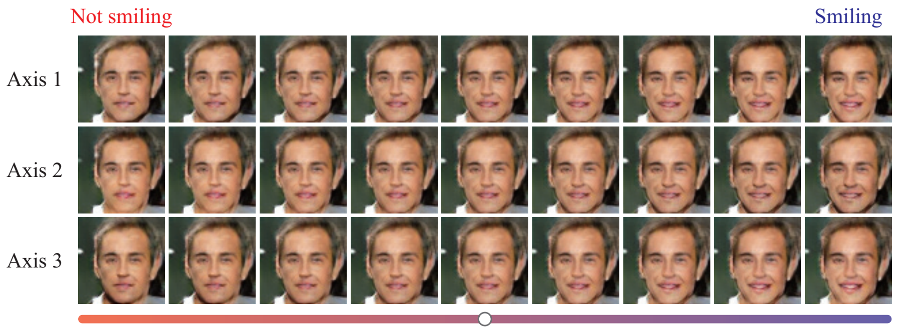
(c) CGAN



(d-1) CFGAN + ten-categorical RB



(d-2) CFGAN + three-dimensional SB-I



(d-3) CFGAN + three-dimensional SB-II

Figure 12. Example results of smiling generation. View of figure is same as that in Figure 8.

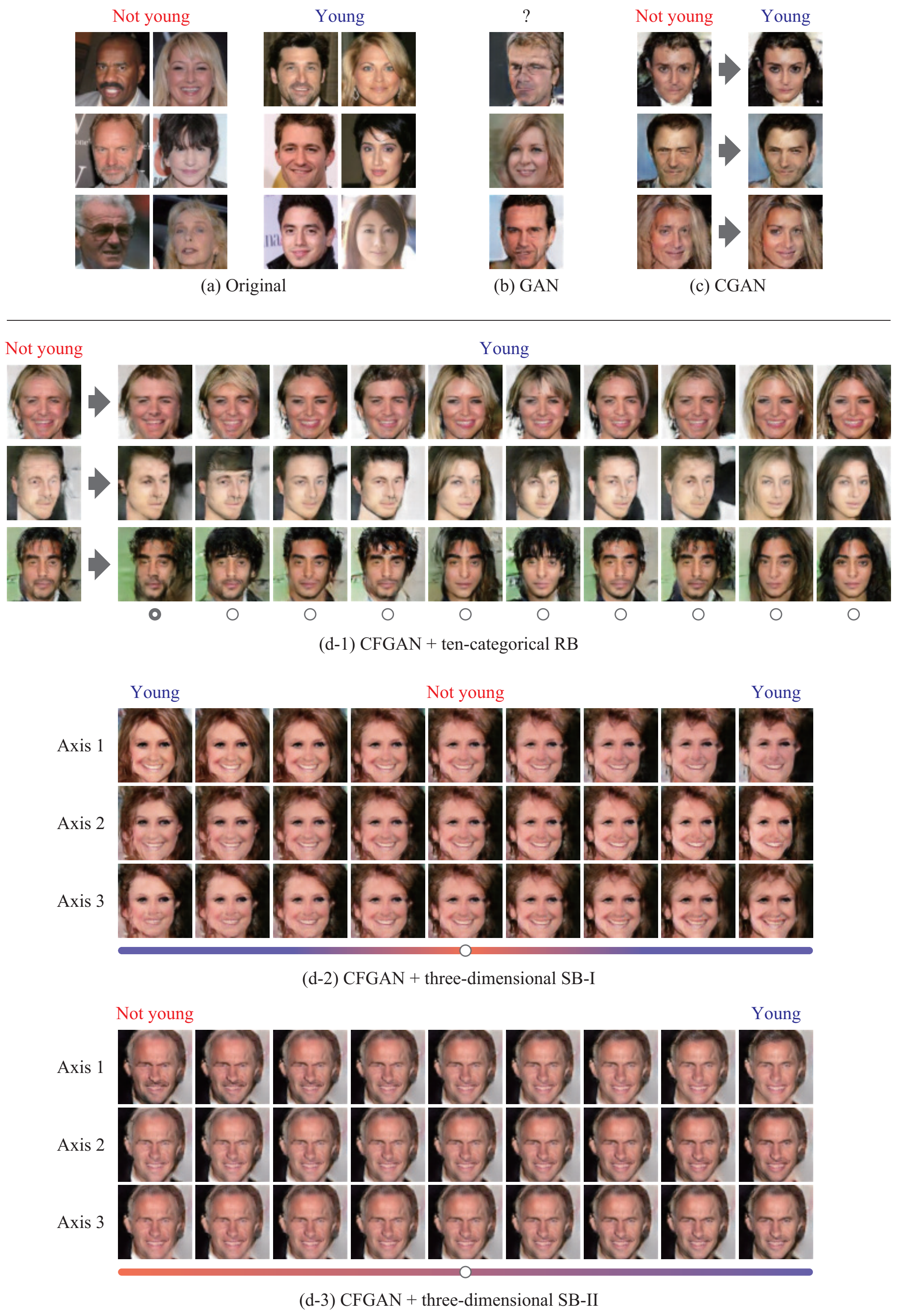
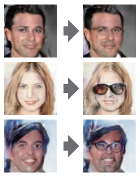
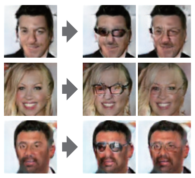
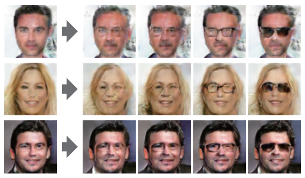


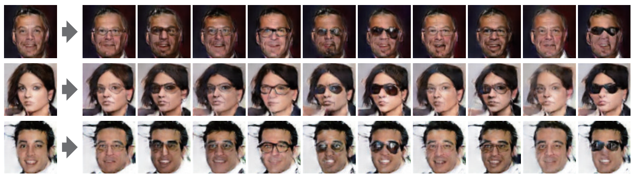
Figure 13. Example results of young generation. View of figure is same as that in Figure 8.

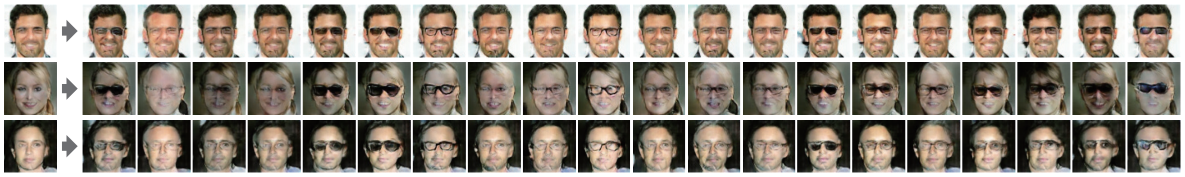
(a) CFGAN + no RB (= GAN)


(b) CFGAN + one-categorical RB (= CGAN)


(c) CFGAN + two-categorical RB


(d) CFGAN + four-categorical RB


(e) CFGAN + ten-categorical RB


(f) CFGAN + 20-categorical RB

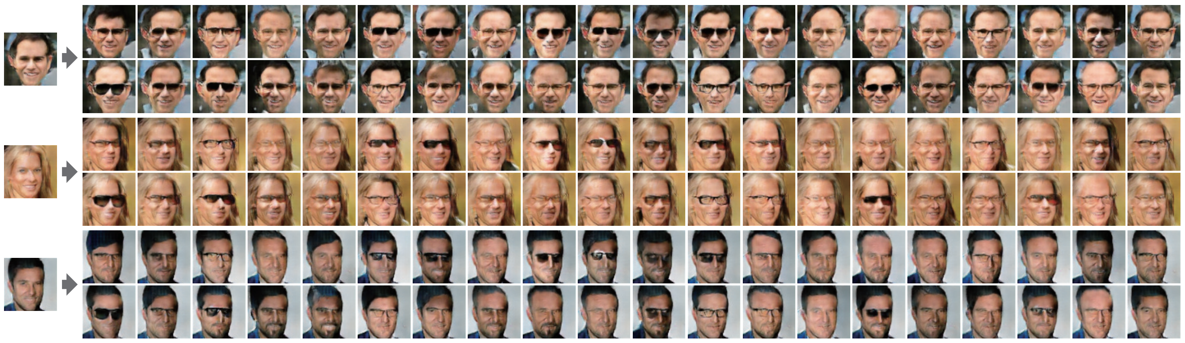
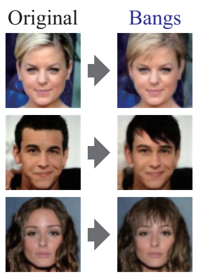
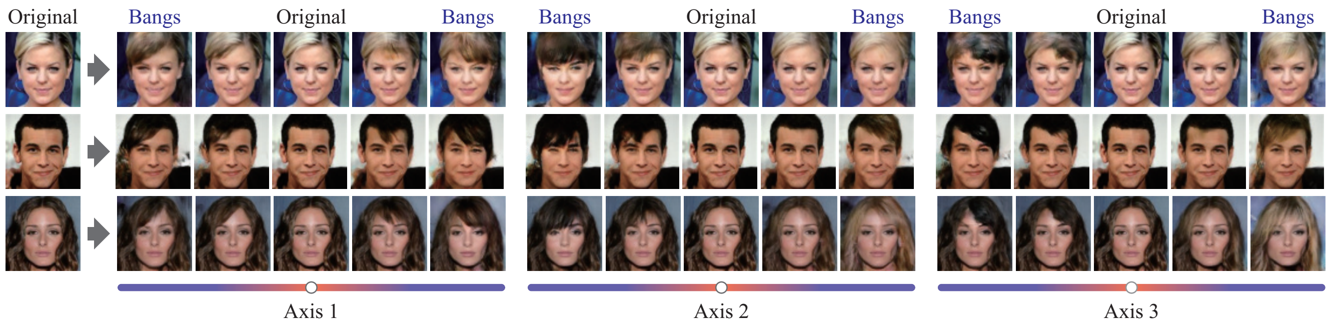

(g) CFGAN + 40-categorical RB

Figure 14. Example results of glasses generation for different categorical RBs

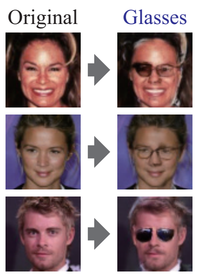


(a) CGAN

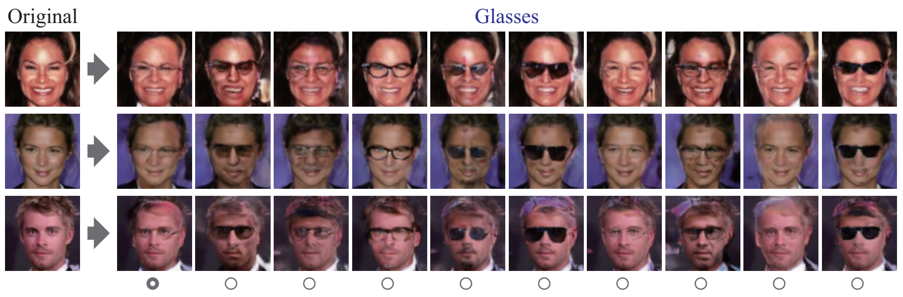


(b) CFGAN + three-dimensional SB-I

Figure 15. Example results of bangs-based image editing. In (a), row shows sample images that contain same z but different y. In (b), row shows sample images that contain same z_i but different z'_a.

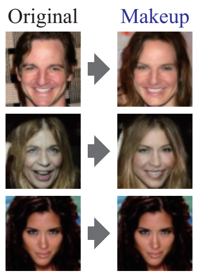


(a) CGAN

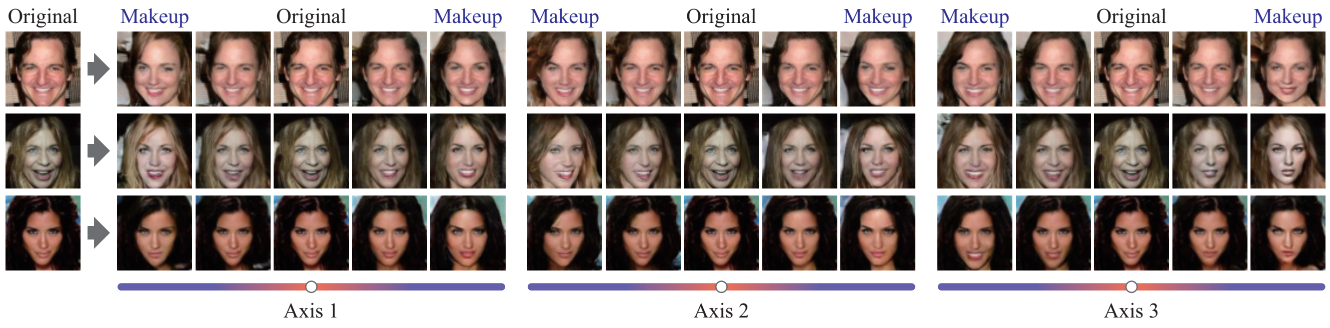


(b) CFGAN + ten-categorical RB

Figure 16. Example results of glasses-based image editing. View of figure is same as that in Figure 15.

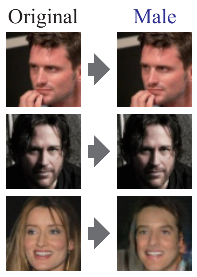


(a) CGAN

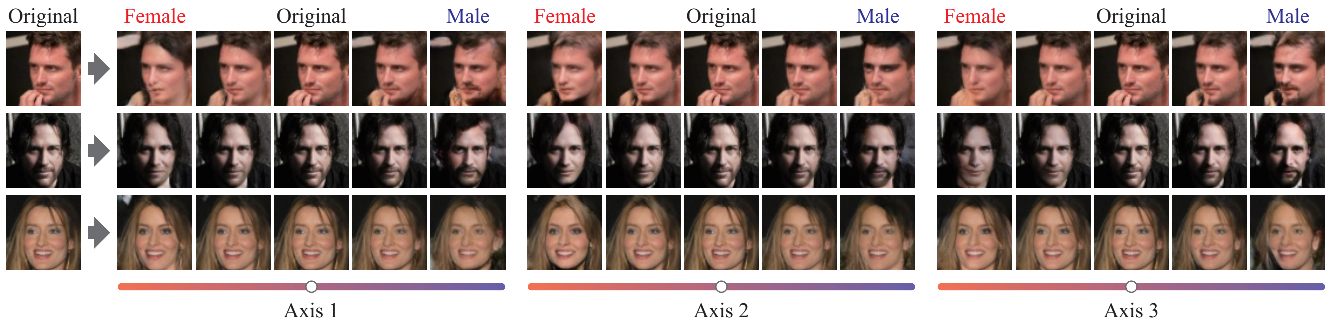


(b) CFGAN + three-dimensional SB-I

Figure 17. Example results of makeup-based image editing. View of figure is same as that in Figure 15.

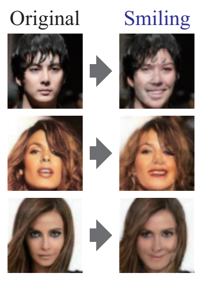


(a) CGAN

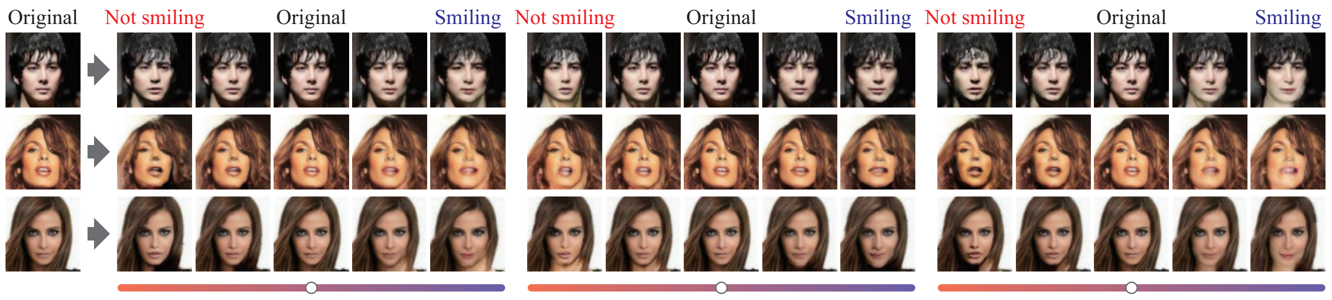


(b) CFGAN + three-dimensional SB-II

Figure 18. Example results of male-based image editing. View of figure is same as that in Figure 15.

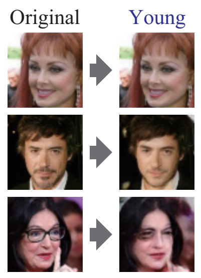


(a) CGAN

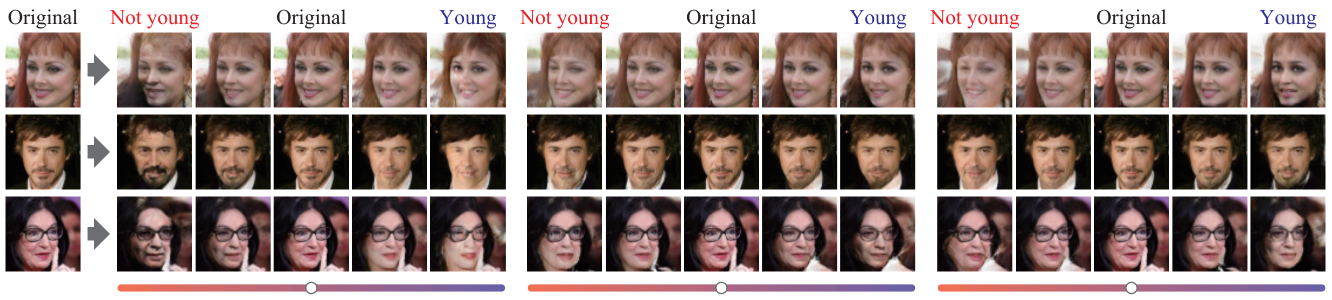


(b) CFGAN + three-dimensional SB-II

Figure 19. Example results of smiling-based image editing. View of figure is same as that in Figure 15.



(a) CGAN



(b) CFGAN + three-dimensional SB-II

Figure 20. Example results of young-based image editing. View of figure is same as that in Figure 15.

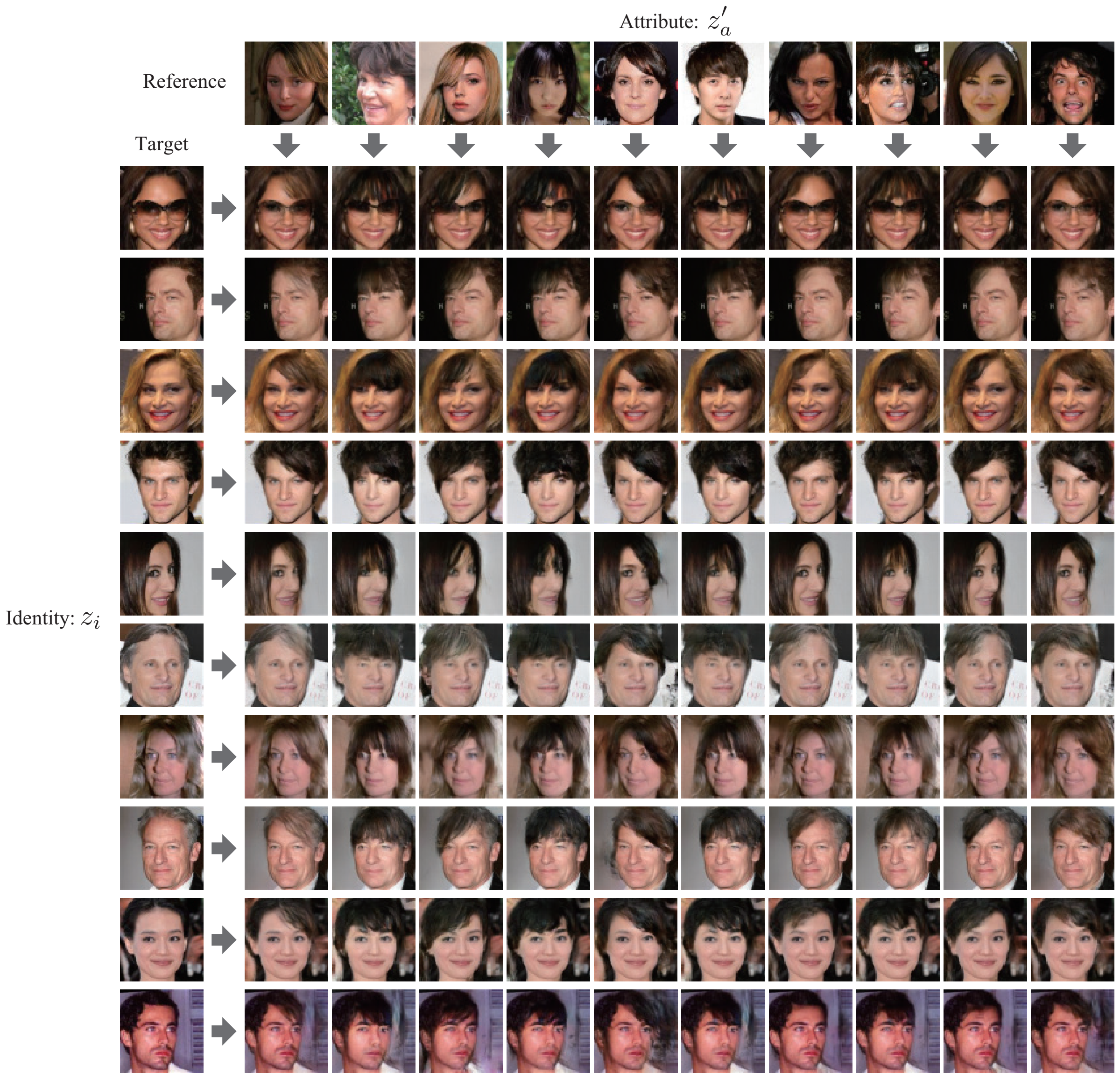


Figure 21. Example results of bangs-based attribute transfer. Images were generated from z_i extracted from first-column images and z'_a extracted from first-row images.

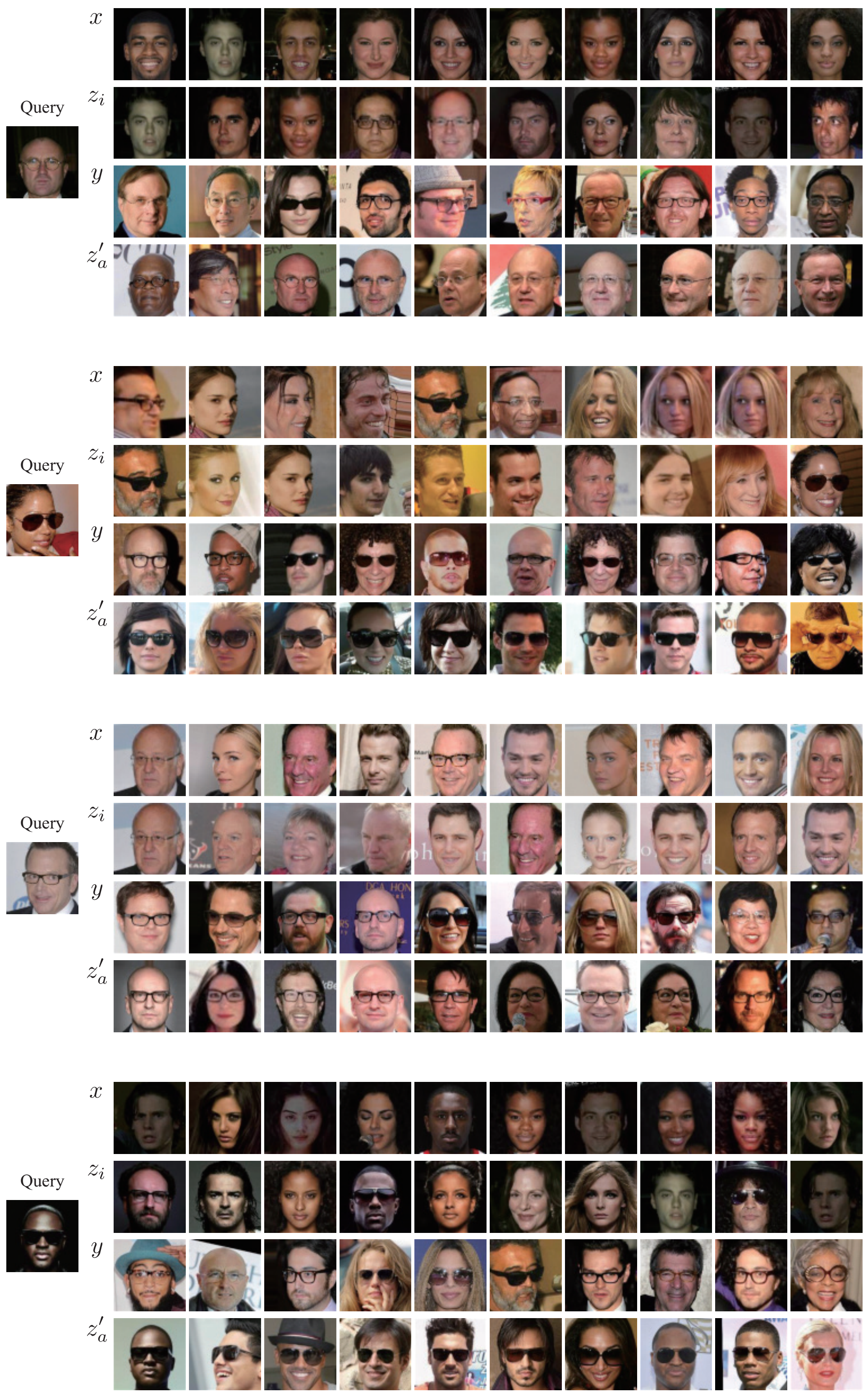


Figure 22. Top ten retrieval results of glasses-based image retrieval. First column shows query images. In other columns, images are ordered from left to right on basis of distance in x, z_i, y, and z'_a.

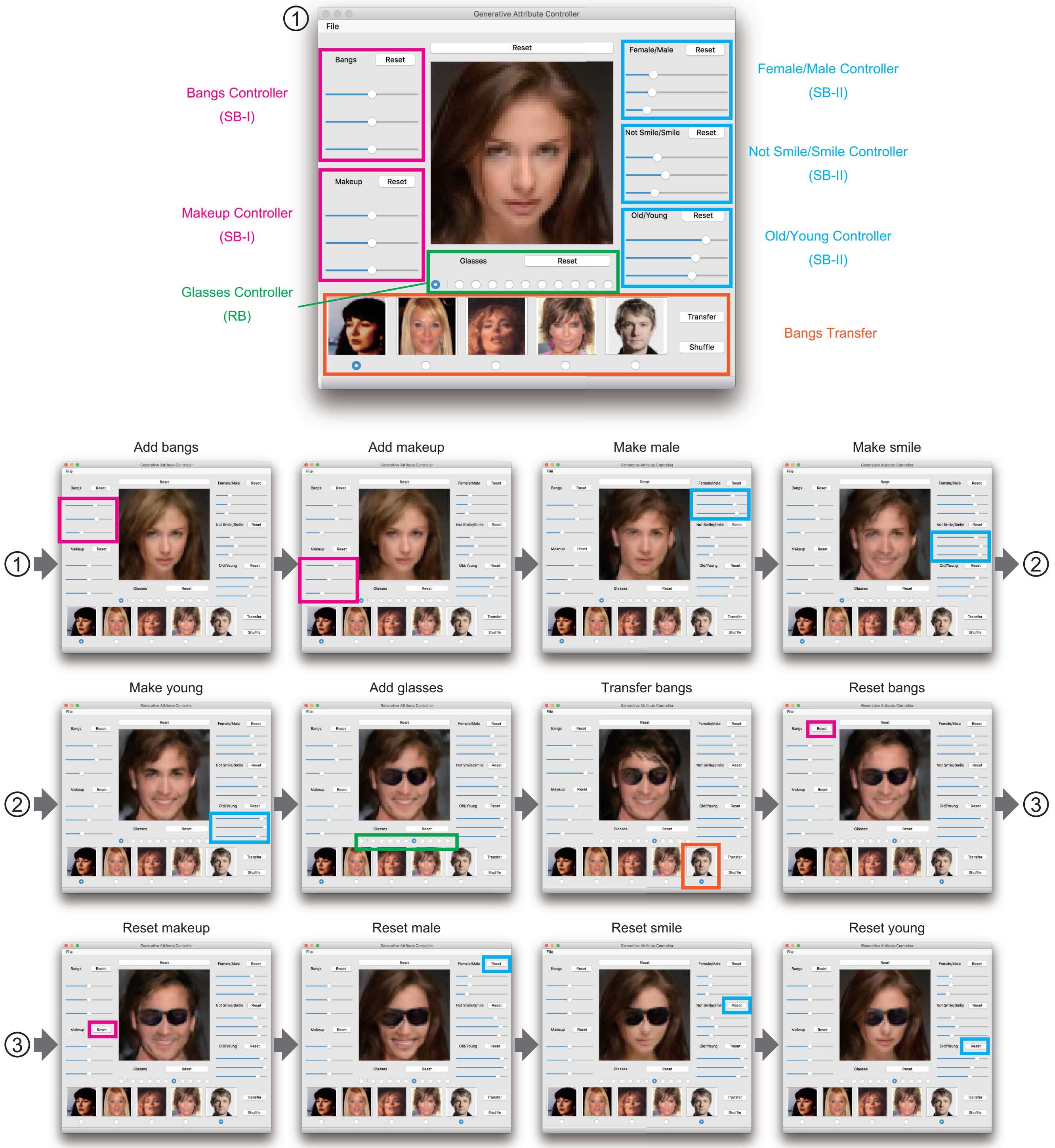


Figure 23. Image-editing examples using interface for GAC with CFGAN. Attributes of image can be edited using typical controllers (slide bars, radio buttons, and reset buttons) while retaining identity.