Image generation & Feature disentanglement

r06946003 湯忠憲

Problem 1. VAE

1. Describe the architecture & implementation details of your model

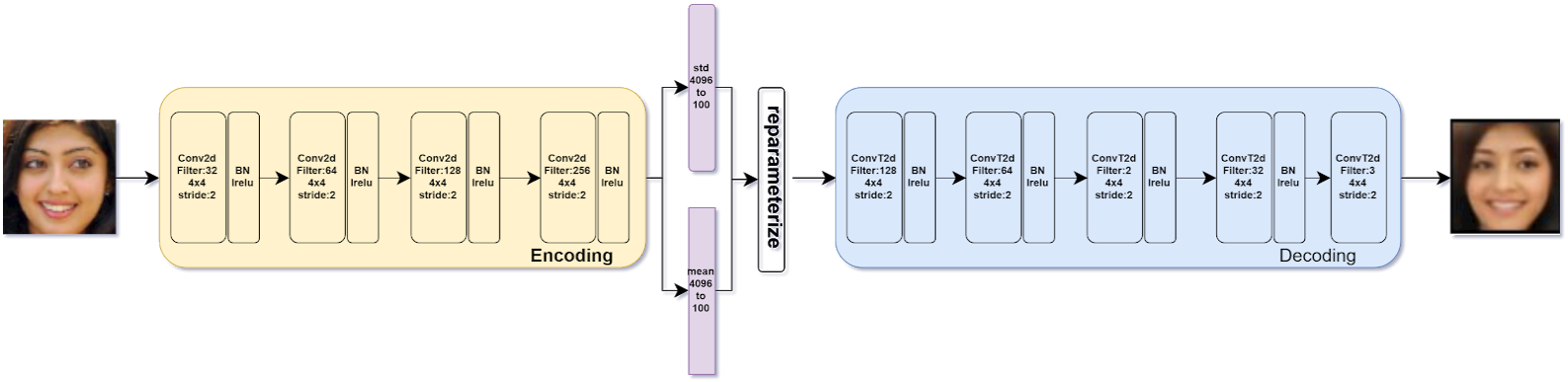

Figure 1. VAE architecture

The VAE model architecture is shown above. For the encoding stage, convolution layers are used to extract the image features. In the middle of architecture, two fully connected layers are learned to predict the mean and log Variance of every dimension of an image in the latent space. In addition, since random sampling is a non-differentiable process, here the reparameterization trick is introduced; that is, given mean and standard deviation the output z is as following in practice: $z = \mu + \varepsilon \times e^{0.5 \times logvar}$. After that, the sampled latent vector (100 dims) is fed into the decoder, which is consist of transpose convolution layers.

Most of the network parameters are shown in Fig 1. The training optimizer is Adam with 0.001 learning rate. Batch size is 64. The loss function of VAE is a combination of mean square error and KL divergence. Since the average operations when computing both MSE and KLD are replaced by "sum" operations, the weights between them are set equal, i.e., $\lambda_{KL} = 1$, which is workable here.

2. Plot the learning curve (reconstruction loss & KL divergence) of your model

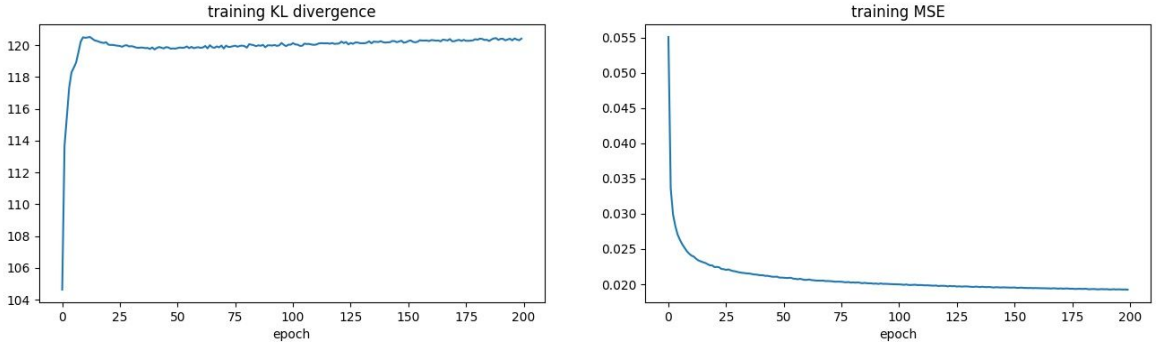

Figure 2. KL divergence and reconstruction loss (MSE)

3. Plot 10 testing images and their reconstructed result of your model and report your testing MSE of the entire test set

Figure 3. Reconstruction for first 10 samples in the testing set.

The test MSE of the entire test set is 0.0197.

4. Plot 32 randomly generated images of your model

Figure 4. Randomly generated samples

5. Visualize the latent space by mapping test images to 2D space (with tSNE) and color them with respect to and attribute of your choice

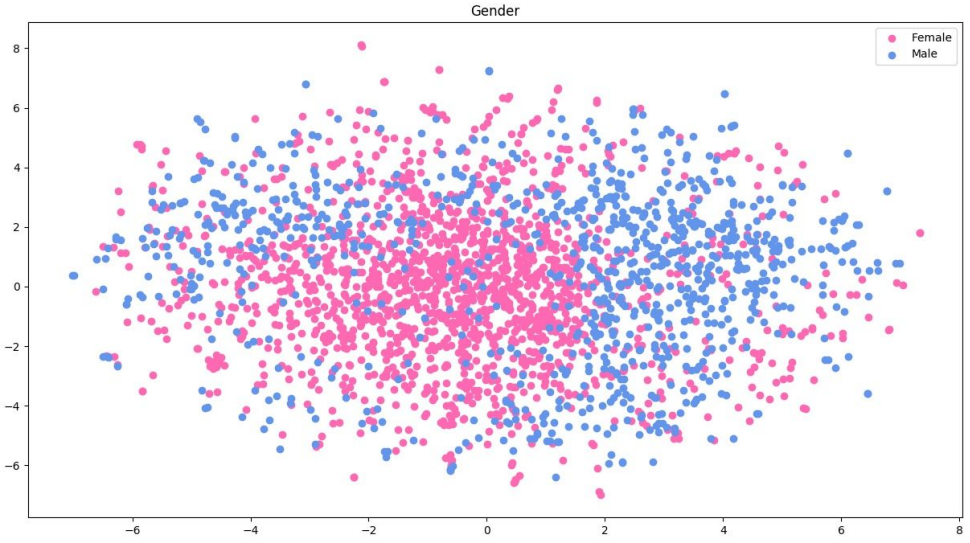

Figure 5. Latent space visualization

6. Discuss what you've observed and learned from implementing VAE
    a. The outputs of VAE are smoother than GAN's output.
    b. Size of latent space did not make a great impact on performance (in my case).
    c. Using "sum" pooling when computing loss somehow could avoid suffering from tuning lambda.
    d. Using sigmoid as output activation reaches similar results.

Problem 2. GAN

1. Describe the architecture & implementation details of your model

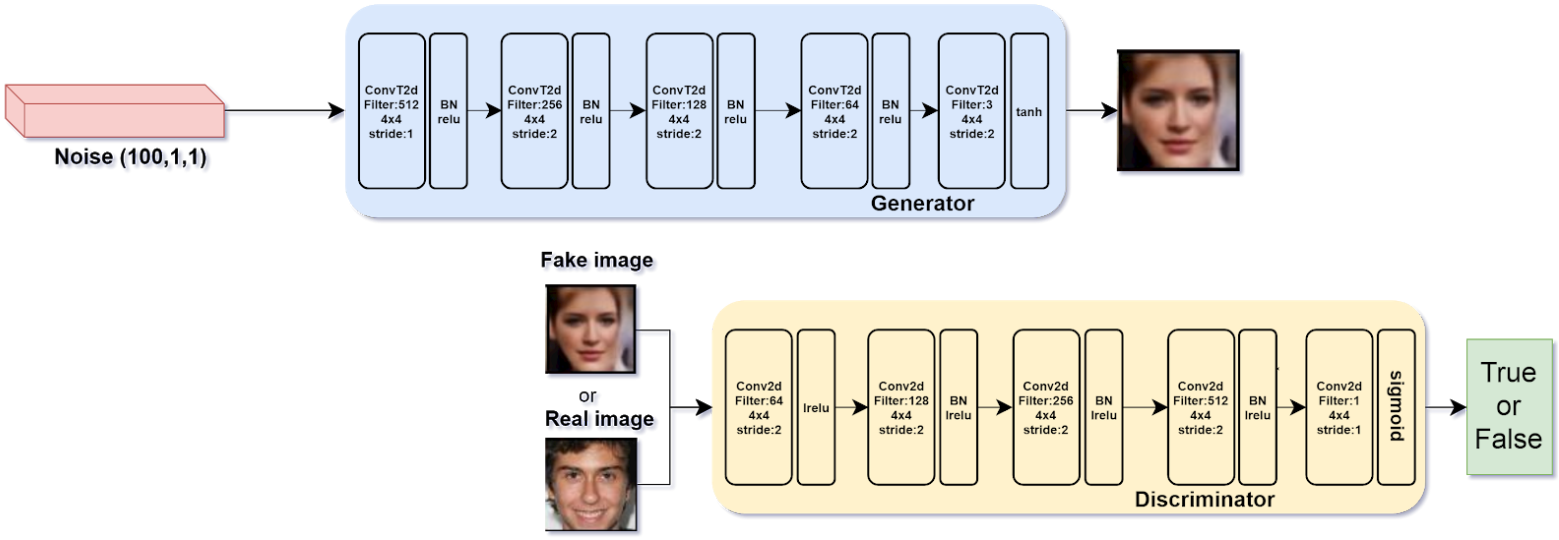

Figure 6. DCGAN architecture

The architecture of DCGAN is shown in Fig 6. In the generator, transpose convolution layers are applied to generate an image from a 100-dim Gaussian random noise. The random noise is reshaped to a cuboid ($100 \times 1 \times 1$) for fitting the first transpose convolution layer. The output of generator goes through a hyperbolic tangent activation function, which makes the pixel values lay in a range from -1 to 1. Dividing the output values by 2 and adding 0.5 could transform values' range from 0 to 1.

For discriminator (yellow part in Fig 6), the model is responsible for telling whether the input image is a real sample dataset. The output range is from 0 to 1 as this is a binary classification problem.

The training optimizer is Adam with 0.0002 learning rate, 0.5 beta1 and 0.999 beta2 for both modules. The loss function is binary cross-entropy here.

2. Plot the learning curve of your model and briefly explain what you think it represents

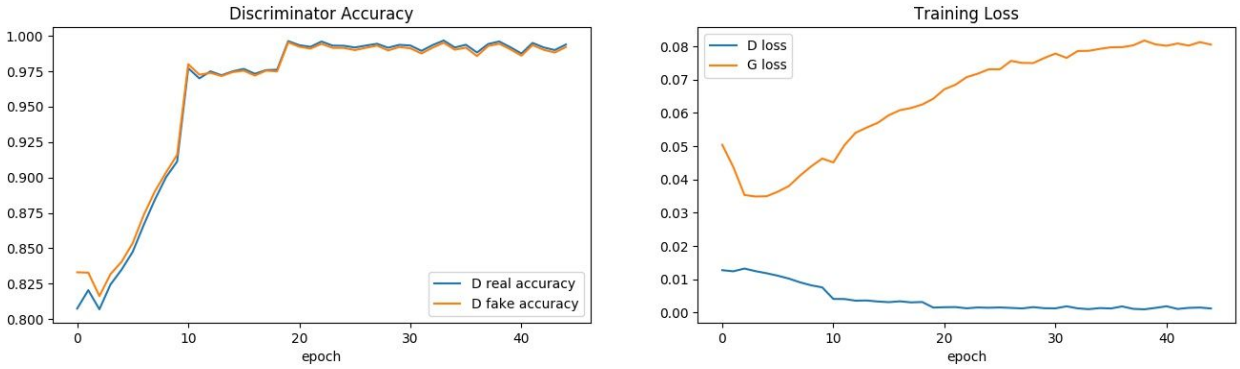

Figure 7. Discriminator accuracy and training loss

The left part of Fig 7 indicates that the discriminator did learn to distinguish between real and fake image. The competitive property of two models could be observed in the right of Fig 7. After 35 epochs, the system reaches the balance. Practically, displaying some randomly generated pictures is a good way to monitor the model performance.

3. Plot 32 randomly generated images of your model

Figure 8. Randomly generated images

4. Discuss what you've observed and learned from implementing GAN
    a. Basically, DCGAN is powerful enough to generate some clear images.
    b. My model seldom generates black people due to the limitation of the dataset.
    c. Flipping the images could make the training data double and enhance the performance.
    d. Random noise from Uniform(-1,1) for training and inference also generates similar results.
5. Compare the difference between the image generated by VAE and GAN, discuss what you've observed
    a. Both of them seldom generate black people.
    b. GAN provides amazing results in terms of the clarity.

c. In VAE, the model loss for backpropagation is from comparison with the real image. On the other hand, in GAN, parameters updating in the generator is not triggered by the data but use the loss of discriminator.
d. Therefore, VAE is a supervised algorithm and GAN is treated as a sort of semi-supervised one.

Problem 3. ACGAN

1. Describe the architecture & implementation details of your model

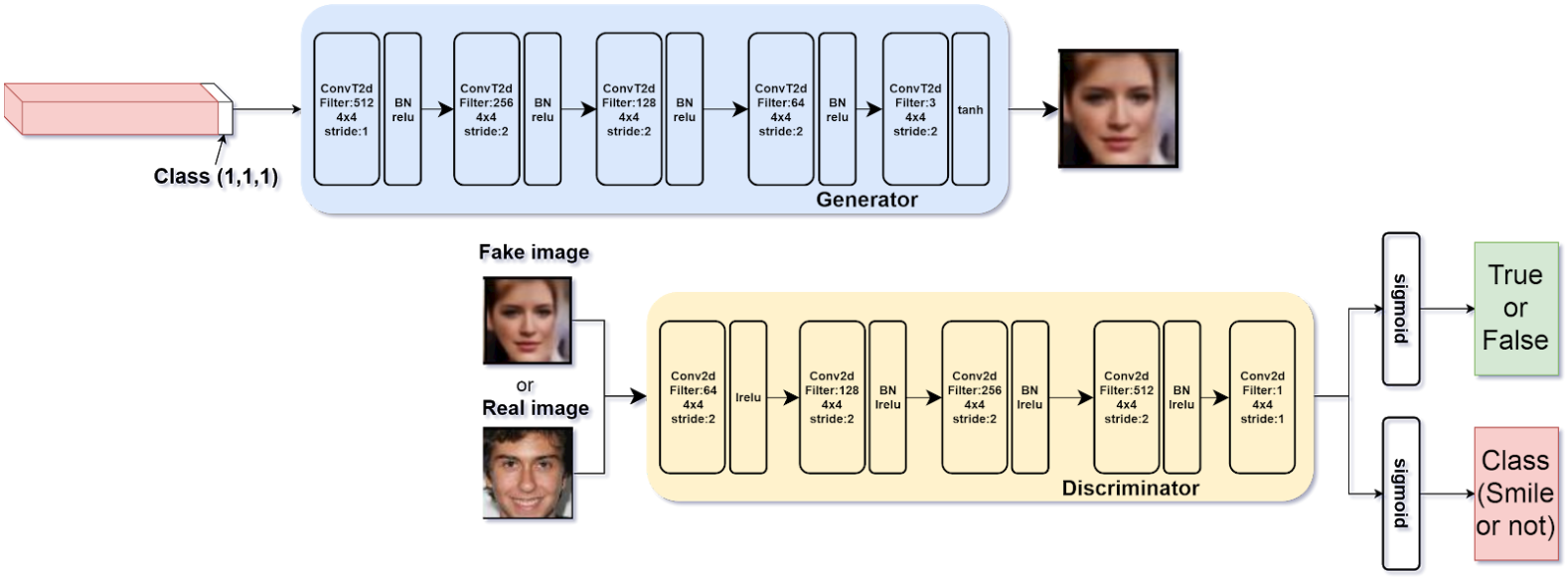

Figure 9. ACGAN architecture

ACGAN is trained with supervised pairs of images and their attributes. The model structure is similar to GAN. Here the noise vector is concatenated with a one-hot vector, which denotes the class (attribute) of an expected output image. For supervised learning, in the discriminator, one more sigmoid output is used for predicting the class of input image.

The training optimizer is Adam with 0.0002 learning rate, 0.5 beta1 and 0.999 beta2 for both modules. The loss function is binary cross-entropy for both sigmoid outputs.

2. Plot the learning curve of your model and briefly explain what you think it represents

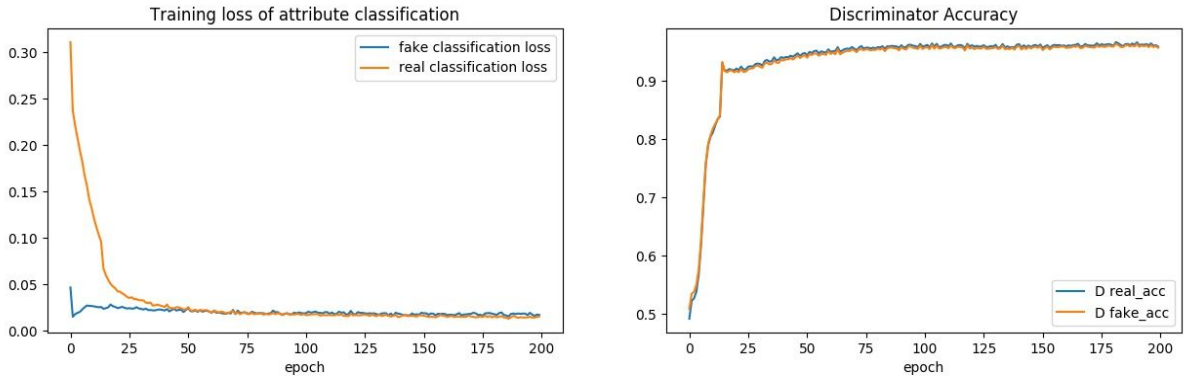

Figure 10. Classification loss and discriminator accuracy

In ACGAN, two kinds of abilities should be taken care of. One is to generate the image with a specific attribute. This could be monitor by the loss of attribute classification shown in the left of Fig 10. Another, same as GAN, the ability to fool discriminator and to distinguish the real/fake images (refers to the right plot in Fig 10).

3. Plot 10 pairs of randomly generated images of your model, each pair generated from the same random vector input but a different attribute. This is to demonstrate your model's ability to disentangle feature of interest.

Figure 11. Feature disentangle (smile)

In this part, ACGAN is trained with *"smile"* attribute. The upper row shows faces with smiles. The second row is "not smile".

Bonus: InfoGAN

1. Describe the architecture & implementation details of your model

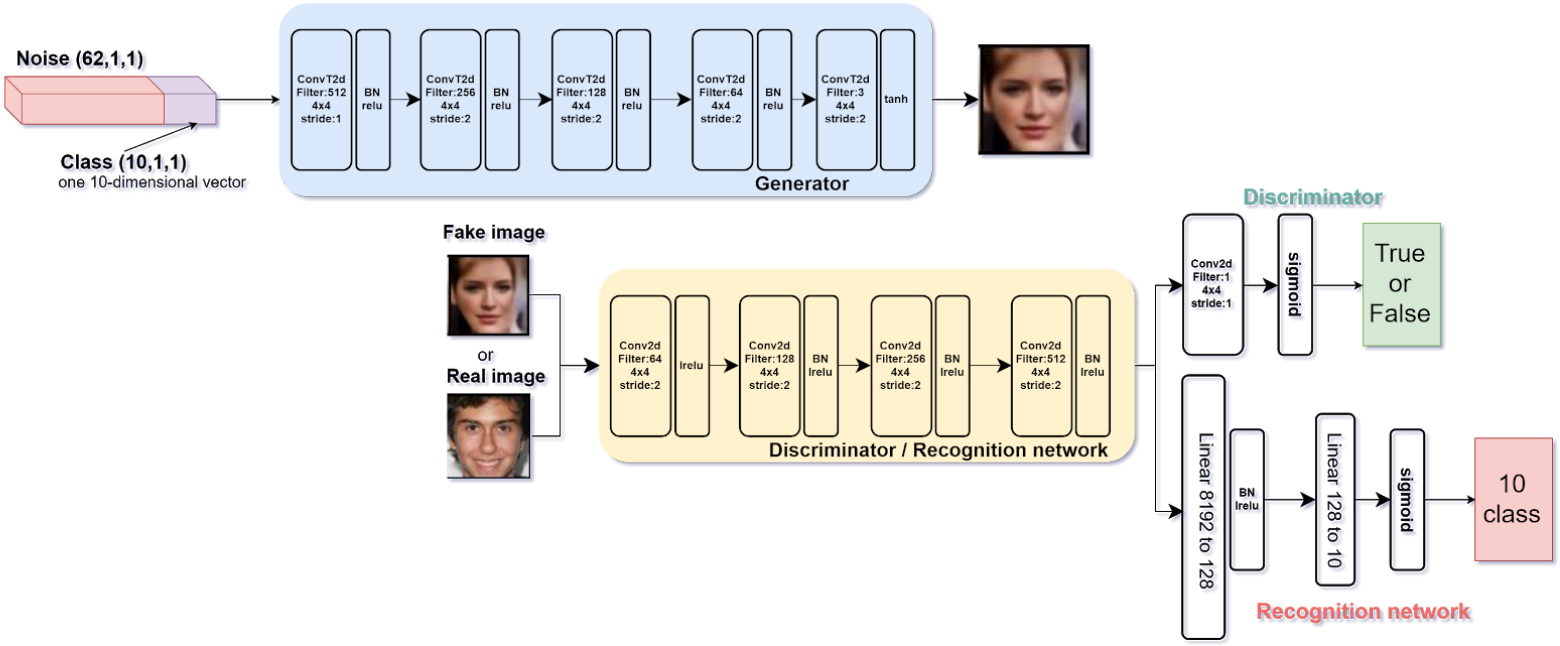

Figure 12. InfoGAN architecture

The network architectures are shown in Fig 12. The discriminator and the recognition network shares most of the network. In the original paper, the author uses whole CelebA dataset to discover some attributes of images. In our task, only 40k images were given, so the input code is simplified here. Categorical codes with several different numbers of the 10-dimensional vector were tried, but nothing astonishing was found. Therefore, for this task, the input vector results in a catenated dimension of 72, which is consist of 1 ten-dimensional categorical code and 62 noise variables. Note that the number of noise variable should adapt for the number of the 10-dimensional code to keep off the ascendancy of itself.

Three training optimizers (for the generator, discriminator and recognition network) are all Adam with learning rates of 0.0002 and betas(0.5, 0.999). The loss functions are binary cross-entropy and cross-entropy for discriminator and recognition network, respectively.

2. Plot the learning curve of your model and briefly explain what you think it represents

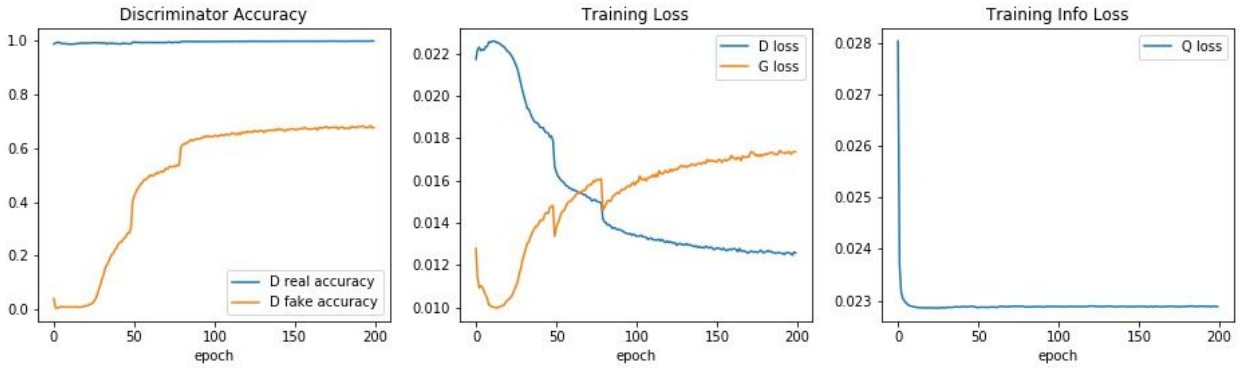

Figure 13. Discriminator accuracy and training loss

Discriminator accuracy and training Q loss reveal capability of the generator to create deceptive images preserved a certain attribute. The middle plot in Fig 13 depicts the competitive relationship between the generator and discriminator.

3. Plot the results

Figure 14. Manipulating latent codes on face images (one 10-dimensional code)

The most distinctive attribute might be the variation in the hairstyle that is easily captured by latent codes. The above figure shows different hair volumes and colors. Other attributes presented in the paper are hard to discover ("pose" might be the second easier one). To get insights into what on earth each code combination represents, results from two 10-dimensional code, i.e., 100 combinations, are shown below. Hairstyle still dominates my vision.

Figure 15. Manipulating latent codes on face images (two 10-dimensional code)