Appendix

1. Detailed Framework Design of Class Association Embedding Learning

The detailed network structure of the discriminator, class-unrelated (E_u) and class-related (E_r) extraction blocks in the encoder, and the decoder are shown in Figure S1, Figure S2, Figure S3 and Figure S4 respectively. The combination of class-unrelated and class-related codes for new samples generation is implemented using MLP which consists of 3 fully connected layers and is presented in Figure S5.

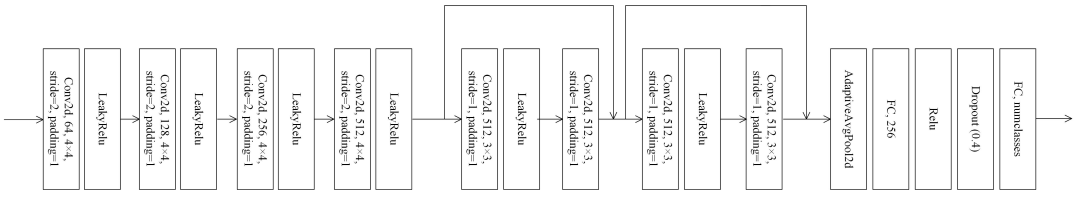

Figure S1: Detailed network structure of the discriminator.

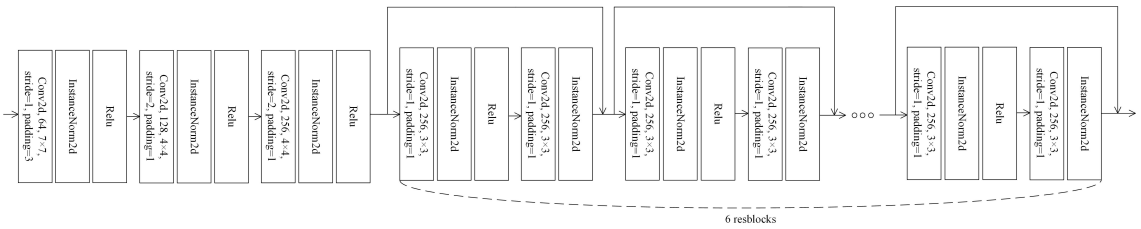

Figure S2: Detailed network structure of the class-unrelated (E_u) code extraction block.

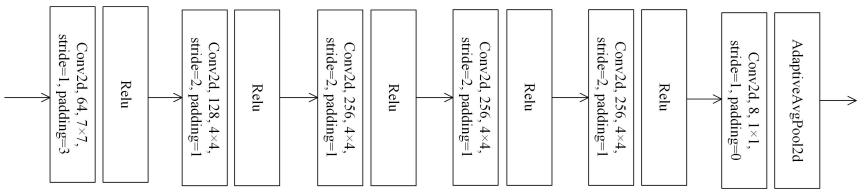

Figure S3: Detailed network structure of the class-related (E_r) code extraction block.

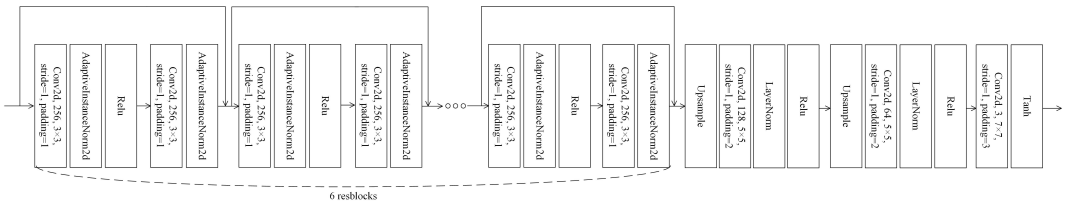

Figure S4: Detailed network structure of the decoder.

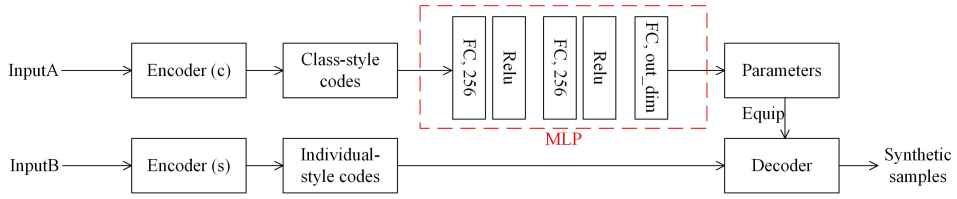

Figure S5: Combination of class-related (class-style) and class-unrelated (individual-style) codes using Multilayer Perceptron Layers.

2. Design of Loss Function for Class Association Embedding Learning

Loss functions are designed to efficiently learn the class-association embedding vectors and achieve the cyclic adversarial learing goals. Let $A : (x_A, y_A)$ be the sample to be credited and $B : (x_B, y_B)$ be its paired sample (whose loss should be credited separately), where y refers to the label of the sample x. Let E_r, E_u and G be the mapping functions of the encoders and decoder described above. c represents the class-related codes while s refers to the class-unrelated codes. The first part of the loss function is the reconstuction loss. The basic reconstruction loss, shown in equation 1, represents the error of the encode-decode loop without any code switch. Moreover, we argue that modifying the class-related code of a sample should not affect its class-unrelated code, and vice versa. Hence, we encode the synthetic sample and compare its class-related and class-unrelated codes with their original counter parts, resulting in the class-wise and sample-wise reconstruction loss denoted in equations 2 and 3, which help with improving the consistency of the code space when training.

$$\mathcal{L}_{recon}^{x_A} = \mathbb{E}_{x_A \sim p(x_A)} \left[\| \, G(E_r(x_A), E_u(x_A)) - x_A \, \|_1 \right] \tag{1}$$

$$\mathcal{L}_{recon}^{c_A} = \mathbb{E}_{c_A \sim p(c_A), s_B \sim q(s_B)} \left[\| \, E_r(G(c_A, s_B)) - c_A \, \|_1 \right] \tag{2}$$

$$\mathcal{L}_{recon}^{s_A} = \mathbb{E}_{c_B \sim p(c_B), s_A \sim q(s_A)} \left[\| \ E_u(G(c_B, s_A)) - s_A \ \|_1 \right] \tag{3}$$

The second part of the loss functin is the cyclic loss as shown in equation 4, which calculates the error of recovering the original sample after the two-round encode-decode cycle with two style switchs. The third and fourth parts are the adversarial and classification loss functions, shown in equations 5 and 6. Where D_r represents the output probability of the sample being real-looking (1 refers to real, 0 fake), and D_c the probability of assigning the correct class (y_A).

$$\mathcal{L}_{cyc}^{x_A} = \mathbb{E}_{x_A \sim p(x_A)} \left[\| \ G(c_A, E_u(G(c_B, s_A))) - x_A \ \|_1 \right] \tag{4}$$

$$\mathcal{L}_{adv1}^{A2B} = \mathbb{E}_{c_B \sim p(c_B), s_A \sim q(s_A)} \left[-log(\frac{exp(D_r(G(c_B, s_A))[1])}{exp(D_r(G(c_B, s_A))[0]) + exp(D_r(G(c_B, s_A))[1])}) \right] \tag{5}$$

$$\mathcal{L}_{cla1}^{A2B} = \mathbb{E}_{c_B \sim p(c_B), s_A \sim q(s_A)} \left[-log(\frac{exp(D_c(G(c_B, s_A))[y_B])}{\sum_{j \in \mathbb{C}} exp(D_c(G(c_B, s_A))[j])}) \right] \tag{6}$$

Combinging the above loss functions, we used the object function in equation 7 to train and update the encoders and decoder.

$$\begin{aligned} \mathcal{L}(E, G) = 10 \times (\mathcal{L}_{recon}^{x_A} + \mathcal{L}_{recon}^{x_B}) + (\mathcal{L}_{recon}^{c_A} + \mathcal{L}_{recon}^{c_B}) + (\mathcal{L}_{recon}^{s_A} + \mathcal{L}_{recon}^{s_B}) \\ +10 \times (\mathcal{L}_{cyc}^{x_A} + \mathcal{L}_{cyc}^{x_B}) + (\mathcal{L}_{adv1}^{A2B} + \mathcal{L}_{adv1}^{B2A}) + (\mathcal{L}_{cla1}^{A2B} + \mathcal{L}_{cla1}^{B2A}) \end{aligned} \tag{7}$$

On the other hand, the parameters in the discriminator D are updated based on equation 8, 9 and 10, which are calculated on each pair of samples together.

$$\begin{aligned} \mathcal{L}_{adv2}^{A2B} = \mathbb{E}_{c_B \sim p(c_B), s_A \sim q(s_A)} \left[-log(\frac{exp(D_r(G(c_B, s_A))[0])}{exp(D_r(G(c_B, s_A))[0]) + exp(D_r(G(c_B, s_A))[1])}) \right] \\ +\mathbb{E}_{x_B \sim p(x_B)} \left[-log(\frac{exp(D_r(x_B)[1])}{exp(D_r(x_B)[0]) + exp(D_r(x_B)[1])}) \right] \end{aligned} \tag{8}$$

$$\mathcal{L}_{cla2}^{A} = \mathbb{E}_{x_A \sim p(x_A)} \left[-log(\frac{exp(D_c(x_A)[y_A])}{\sum_{j \in \mathbb{C}} exp(D_c(x_A)[j])}) \right] \tag{9}$$

$$\mathcal{L}(D) = (\mathcal{L}_{adv2}^{A2B} + \mathcal{L}_{adv2}^{B2A}) + 2 \times (\mathcal{L}_{cla2}^{A} + \mathcal{L}_{cla2}^{B}) \tag{10}$$

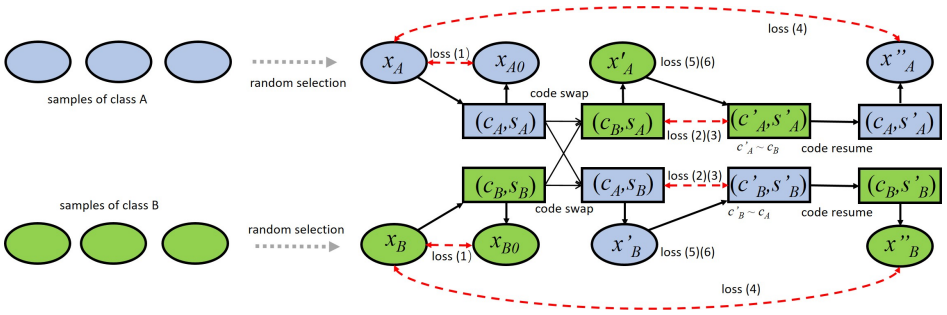

Figure S6: The detailed training schema for Class Association Embedding Learning.

3. Detailed Training Schema

The detailed training schema of class association embedding learning is presented in Figure S6. The numbers in parentheses showed the portion of loss functions given by the corresponding equation presented above. In order to learn the efficient representation of the class-realted style in a unified domain, we randomly select two images from different classes from the training set for pairing. Without loss of generality, for two paired samples x_A and x_B, the two classes can be denoted by y_A and y_B, respectively. Multi-class tasks can be also learned in a 1-vs-1 manner. The paired samples (x_A and x_B) are fed into the same encoder to generate individual/class-unrelated codes (s_A and s_B) and class-related codes (c_A and c_B) respectively. Decoding (c_A, s_A) and (c_B, s_B) resembles x_A and x_B. A code-swith scheme is performed so that the combination (c_B, s_A) and (c_A, s_B) lead to two synthetic samples x'_A and x'_B with switched class assignments $y'_A = y_B$ and $y'_B = y_A$. Both samples are then re-encoded as (c'_A, s'_A) and (c'_B, s'_B) with $c'_A \sim c_B$ and $c'_B \sim c_A$. A second-round combination is peformed and thus (c_A, s'_A) and (c_B, s'_B) are decoded into $x''_A \sim x_A$ and $x''_B \sim x_B$, respectively. During random paired training process, without loss of generality, an sample can be randomly (or enumeratedly when the data size is small) paired with many samples of different classes to generate a large number of class-individual style combinations. After enough iterations of training, class-related and individual/class-unrelated subspaces are separated successfully.