Detach and Adapt: Learning Cross-Domain Disentangled Deep Representation Supplementary

1. Additional Experiments

We provide additional experiments of disentangled representation learning with the attribute of *smiling*, on face images for Sketch→Photo. Fig. 1 shows the disentangled and manipulated results, and those of cross-domain conditional image translation.

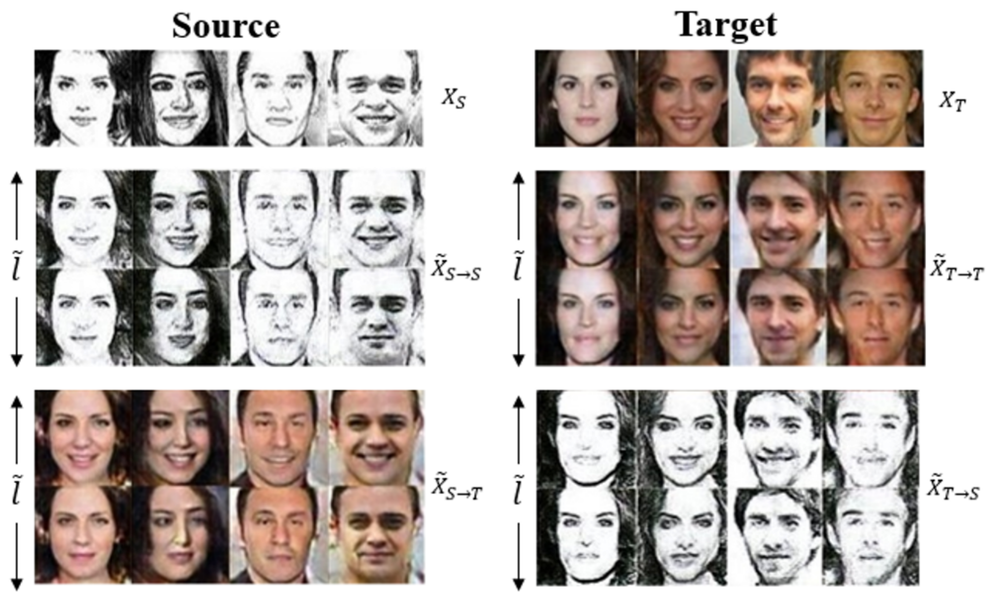

Figure 1: Cross-domain conditional image translation for facial images: Sketch → Photo with $\tilde{l}$ as *smiling*.

2. Implementation Details

Learning rate: The learning rates of all cases are fixed as 10^{-4}.
Update frequency: For each iteration, we found that satisfactory results were obtained with each component being updated once.
Batch size: The batch size of 64 is used for scenarios of *digits*, while for scenarios of *faces* and *scenes* batch size of 8 is used.
Weight: For the objective functions listed in our paper, we adjust weights for each term in order to balance between each component. First, the gradient decent steps for learning CDRD, i.e. Equation (4) in the main paper, can be rewritten as:

$$\begin{aligned} \theta_G &\xleftarrow{+} -\Delta_{\theta_G}(-\lambda_{adv}\mathcal{L}_{adv} + \lambda_{dis}\mathcal{L}_{dis}) \\ \theta_D &\xleftarrow{+} -\Delta_{\theta_D}(\lambda_{adv}\mathcal{L}_{adv} + \lambda_{dis}\mathcal{L}_{dis}) \end{aligned} \tag{1}$$

The ratio of λ_{vae} to λ_{adv} is $1:1$ for scenarios of *digits*, and $1:0.5$ for scenarios of *faces* and *scenes*.

Next, the objective functions of learning VAE of E-CDRD can be rewritten as follows:

$$\mathcal{L}_{vae}^{S} = \lambda_{perc}\|\Phi(X_S) - \Phi(\tilde{X}_{S\to S})\|_F^2 + KL(q_S(z_S|X_S)||p(z))$$

$$\mathcal{L}_{vae}^{T} = \lambda_{perc}\|\Phi(X_T) - \Phi(\tilde{X}_{T\to T})\|_F^2 + KL(q_T(z_T|X_T)||p(z)).$$

In our experiments, we found that larger weights, i.e. λ_{perc}, are preferable for the perceptual loss terms of *faces* and *scenes*. This allowed us to preserve the identity and perceptual information, respectively.

The gradient decent steps for learning E-CDRD, i.e. Equation (11) in the main paper, can be rewritten as follows:

$$\begin{aligned} \theta_E &\xleftarrow{+} -\Delta_{\theta_E}(\lambda_{vae}\mathcal{L}_{vae}) \\ \theta_G &\xleftarrow{+} -\Delta_{\theta_G}(\lambda_{vae}\mathcal{L}_{vae} - \lambda_{adv}\mathcal{L}_{adv} + \lambda_{dis}\mathcal{L}_{dis}) \\ \theta_D &\xleftarrow{+} -\Delta_{\theta_D}(\lambda_{adv}\mathcal{L}_{adv} + \lambda_{dis}\mathcal{L}_{dis}). \end{aligned} \tag{2}$$

The ratio of λ_{vae} to λ_{adv} is $1:1$ for all scenarios. For the ratio of λ_{adv} to λ_{dis}, it is $1:1$ for scenarios of *digits*, and $1:0.5$ for scenarios of *faces* and *scenes*.

Network Architecture. The network architectures for different experimental scenarios are listed in Tables 1, 2 and 3, respectively. The slope of Leaky ReLU in our model is set as 0.2.

Table 1: The network architecture of our CDRD for *digits*. (* indicates parallel layers.)

Generator			
	Layer	Activation Size	Activ. Fun.
Input	-	$256 + 10$	-
G_C	FC	$2 \cdot 2 \cdot 1024$	Leaky ReLU
	3×3 Conv.	$5 \times 5 \times 512$	Leaky ReLU
	3×3 Conv.	$12 \times 12 \times 256$	Leaky ReLU
G_S/G_T	3×3 Conv.	$25 \times 25 \times 128$	Leaky ReLU
	4×4 Conv.	$28 \times 28 \times 1$	Tanh
Discriminator			
Input	-	$28 \times 28 \times 1$	-
D_S/D_T	5×5 Conv.	$28 \times 28 \times 20$	Leaky ReLU
D_C	5×5 Conv.	$28 \times 28 \times 50$	Leaky ReLU
	5×5 Conv.	$28 \times 28 \times 500$	Leaky ReLU
	FC	500	Sigmoid
	*FC: Real/Fake	2	Softmax
	*FC: Class	10	Softmax

Optimizer. ADAM [1] optimizer is chosen to train our model, with β_1 and β_2 set as 0.5 and 0.999, respectively.

References

[1] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

Table 2: The network architecture of our CDRD for *faces* and *scenes*. (* indicates parallel layers.)

	Layer	Activation Size	Activ. Fun.
Generator			
Input	-	$512 + 1$	-
G_C	FC	$4 \cdot 4 \cdot 1024$	Leaky ReLU
	4×4 Conv.	$8 \times 8 \times 512$	Leaky ReLU
	4×4 Conv.	$16 \times 16 \times 256$	Leaky ReLU
	4×4 Conv.	$32 \times 32 \times 128$	Leaky ReLU
G_S/G_T	4×4 Conv.	$64 \times 64 \times 32$	Leaky ReLU
	3×3 Conv.	$64 \times 64 \times 3$	Tanh
Discriminator			
Input	-	$64 \times 64 \times 3(or1)$	-
D_S/D_T	5×5 Conv.	$32 \times 32 \times 64$	Leaky ReLU
	5×5 Conv.	$16 \times 16 \times 128$	Leaky ReLU
D_C	5×5 Conv.	$8 \times 8 \times 256$	Leaky ReLU
	3×3 Conv.	$4 \times 4 \times 512$	Leaky ReLU
	FC	2048	Sigmoid
	*FC: Real/Fake	2	Softmax
	*FC: Class	2	Softmax

Table 3: The network architecture of our E-CDRD for *faces* and *scenes*. (*-indicate parallel layers.)

Component	Layer	Activation Size	Activ. Fun.
Encoder			
Input	-	$64 \times 64 \times 3(or1)$	-
E_S/E_T	5×5 Conv.	$32 \times 32 \times 64$	Leaky ReLU
	5×5 Conv.	$16 \times 16 \times 128$	Leaky ReLU
E_C	5×5 Conv.	$8 \times 8 \times 256$	Leaky ReLU
	3×3 Conv.	$4 \times 4 \times 512$	Leaky ReLU
	FC	2048	Leaky ReLU
	FC	512	Tanh
Generator			
Input	-	$512 + 1$	
G_C	FC	$4 \cdot 4 \cdot 1024$	Leaky ReLU
	4×4 Conv.	$8 \times 8 \times 512$	Leaky ReLU
	4×4 Conv.	$16 \times 16 \times 256$	Leaky ReLU
	4×4 Conv.	$32 \times 32 \times 128$	Leaky ReLU
G_S/G_T	4×4 Conv.	$64 \times 64 \times 32$	Leaky ReLU
	3×3 Conv.	$64 \times 64 \times 3$	Tanh
Discriminator			
Input	-	$64 \times 64 \times 3(or1)$	
D_S/D_T	5×5 Conv.	$32 \times 32 \times 64$	Leaky ReLU
	5×5 Conv.	$16 \times 16 \times 128$	Leaky ReLU
D_C	5×5 Conv.	$8 \times 8 \times 256$	Leaky ReLU
	3×3 Conv.	$4 \times 4 \times 512$	Leaky ReLU
	Fully-connected	2048	Sigmoid
	*FC: Real/Fake	2	Softmax
	*FC: Class	2	Softmax