Duplex Generative Adversarial Network for Unsupervised Domain Adaptation (Supplementary Material)

1. Network Architectures

1.1. Experiments on Digit Classification

In the experiment on digit classification, we use the same architecture as the state-of-the-art methods. Concretely, the state-of-the-art method of MNIST $\leftrightarrow$ USPS and SVHN $\rightarrow$ MNIST is UNIT [3], and that of MNIST $\rightarrow$ SVHN is ATDA [6], so we use the same architectures as the original papers, detailed in Table 1 and Table 2.

Encoder	**Architecture**
1st layer	CONV-(N32,K5,S1), PReLU, MAX-POOL-(K2,S2)
2nd layer	CONV-(N64,K5,S1), PReLU, MAX-POOL-(K2,S2)
3rd layer	CONV-(N128,K7,S1), PReLU
4th layer	CONV-(N256,K1,S1), PReLU
Classifier	**Architecture**
1st layer	FC-(N10)
Generator	**Architecture**
1st layer	DECONV-(N128,K4,S4), BN, PReLU
2nd layer	DECONV-(N64,K3,S2), BN, PReLU
3rd layer	DECONV-(N32,K3,S2), BN, PReLU
4th layer	DECONV-(N1,K6,S1), TanH
Discriminators	**Architecture**
1st layer	CONV-(N32,K5,S1), PReLU, MAX-POOL-(K2,S2)
2nd layer	CONV-(N64,K5,S1), PReLU, MAX-POOL-(K2,S2)
3rd layer	CONV-(N512,K4,S1), PReLU
4th layer	FC-(N11)

Table 1: The architectures of our DupGAN used in MNIST $\leftrightarrow$ USPS. In each layer, the (N, K, S) stand for number of output channels, kernel size, and stride, respectively.

1.2. Experiments on Object Recognition

In the experiment on object recognition, we use the same architecture as the state-of-the-art method DRCN [1], i.e., AlexNet [2] is used as the architecture of the encoder and discriminators, and fc6-conv5-conv4 of AlexNet [2] is as the architecture of the generator. For the training set is too small to train such large model as AlexNet [2] from scratch, we use the pre-trained AlexNet [2] with ImageNet [5], fix the layers conv1-conv3, finetune conv4-fc7, and train the classifier layer fc8 from scratch, as in DAN [4] and DRCN [1]. The architectures are detailed in Table 3.

Encoder	**Architecture**
1st layer	CONV-(N64,K5,S2), BN, LeakyReLU
2nd layer	CONV-(N128,K5,S2), BN, LeakyReLU
3rd layer	CONV-(N256,K5,S2), BN, LeakyReLU
4th layer	CONV-(N512,K4,S1), BN, LeakyReLU
Classifier	**Architecture**
1st layer	FC-(N10)
Generator	**Architecture**
1st layer	DECONV-(N256,K4,S4), BN, LeakyReLU
2nd layer	DECONV-(N128,K4,S2), BN, LeakyReLU
3rd layer	DECONV-(N64,K4,S2), BN, LeakyReLU
4th layer	DECONV-(N3,K4,S2), TanH
Discriminators	**Architecture**
1st layer	CONV-(N64,K5,S1), BN, LeakyReLU
2nd layer	CONV-(N128,K5,S1), BN, LeakyReLU
3rd layer	CONV-(N256,K5,S1), BN, LeakyReLU
4th layer	CONV-(N512,K4,S1), BN, LeakyReLU
5th layer	FC-(N11)

(a) SVHN $\rightarrow$ MNIST

Encoder	**Architecture**
1st layer	CONV-(N64,K5,S1), ReLU, MAX-POOL-(K3,S2)
2nd layer	CONV-(N64,K5,S1), ReLU, MAX-POOL-(K3,S2)
3rd layer	CONV-(N128,K5,S1), ReLU
4th layer	CONV-(N512,K4,S1), ReLU
Classifier	**Architecture**
1st layer	CONV-(N3072,K4,S1), BN, ReLU
2nd layer	CONV-(N2048,K1,S1), ReLU
3rd layer	FC-(N10)
Generator	**Architecture**
1st layer	DECONV-(N256,K4,S4), BN, PReLU
2nd layer	DECONV-(N128,K4,S2), BN, PReLU
3rd layer	DECONV-(N64,K4,S2), BN, PReLU
4th layer	DECONV-(N3,K4,S2), TanH
Discriminators	**Architecture**
1st layer	CONV-(N64,K5,S1), ReLU, MAX-POOL-(K3,S2)
2nd layer	CONV-(N64,K5,S1), ReLU, MAX-POOL-(K3,S2)
3rd layer	CONV-(N128,K5,S1), ReLU
4th layer	CONV-(N3072,K4,S1), BN, ReLU
5th layer	CONV-(N2048,K1,S1), ReLU
6th layer	FC-(N11)

(b) MNIST $\rightarrow$ SVHN

Table 2: The architectures of our DupGAN used in MNIST $\leftrightarrow$ SVHN. In each layer, the (N, K, S) stand for number of output channels, kernel size, and stride, respectively. The first 4 layers of the discriminators in MNIST $\rightarrow$ SVHN are weight-sharing for two domains.

References

[1] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, pages 597–613. Springer, 2016.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the IEEE Conference on Advances in Neural*

Encoder	Architecture
1st layer	CONV-(N96,K11,S4), ReLU, LRN, MAX-POOL-(K3,S2)
2nd layer	CONV-(N256,K5,S1), ReLU, LRN, MAX-POOL-(K3,S2)
3rd layer	CONV-(N384,K3,S1), ReLU
4th layer	CONV-(N384,K3,S1), ReLU
5th layer	CONV-(N256,K3,S1), ReLU, MAX-POOL-(K3,S2)
Classifier	**Architecture**
1st layer	FC-(N4096), ReLU
2nd layer	FC-(N4096), ReLU
3rd layer	FC-(N31)
Generator	**Architecture**
1st layer	DECONV-(N256,K6,S6), ReLU
2nd layer	DECONV-(N384,K3,S2), ReLU
3rd layer	DECONV-(N384,K3,S1), ReLU
Discriminators	**Architecture**
1st layer	CONV-(N96,K11,S4), ReLU, LRN, MAX-POOL-(K3,S2)
2nd layer	CONV-(N256,K5,S1), ReLU, LRN, MAX-POOL-(K3,S2)
3rd layer	CONV-(N384,K3,S1), ReLU
4th layer	CONV-(N384,K3,S1), ReLU
5th layer	CONV-(N256,K3,S1), ReLU, MAX-POOL-(K3,S2)
6th layer	FC-(N4096), ReLU
7th layer	FC-(N4096), ReLU
8th layer	FC-(N32)

Table 3: The architectures of our DupGAN used in object recognition. In each layer, the (N, K, S) stand for number of output channels, kernel size, and stride, respectively. The first 6 layers of discriminators are weight-sharing for two domains.

Information Processing Systems (NIPS), pages 1097–1105, 2012.

[3] M. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*, 2017.

[4] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *Proceedings of the IEEE International Conference on Machine learning (ICML)*, pages 97–105, 2015.

[5] O. Russakovsky, J. Deng, H. Su, and *et al*. Imagenet large scale visual recognition challenge. *IEEE International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[6] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. *arXiv preprint arXiv:1702.08400*, 2017.