# Generative Adversarial Guided Learning for Domain Adaptation (Supplementary Materials)

Kai-Ya Wei
jakc4103@gmail.com
Chiou-Ting Hsu
cthsu@cs.nthu.edu.tw

Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan

## 1 CNN Architecture

Table 1 and Table 2 show the CNN architecture of classification model $D$, which is the same as in [1]. The numbers of convolution kernels (A, B, C) are (64, 64, 64) for the small CNN and (96, 192, 192) for the large CNN. Table 3 shows the architecture of generator $G$. All activation functions after convolution and dense layer are leaky ReLU with $\alpha = 0.1$ if not specified in the tables. Batch normalization is applied before activation in $D_f$, $D_y$, and $G$.

| Layer type | Output size | model indice |
|---|---|---|
| Input image | $32 \times 32 \times 3$ | $D_f$ |
| Instance normalization | $32 \times 32 \times 3$ | $D_f$ |
| Conv. $3 \times 3 \times$ A, stride 1, pad 1 | $32 \times 32 \times$ A | $D_f$ |
| Conv. $3 \times 3 \times$ A, stride 1, pad 1 | $32 \times 32 \times$ A | $D_f$ |
| Conv. $3 \times 3 \times$ A, stride 1, pad 1 | $32 \times 32 \times$ A | $D_f$ |
| Max-pooling $2 \times 2$, stride 2 | $16 \times 16 \times$ A | $D_f$ |
| Dropout ,rate=0.5 | $16 \times 16 \times$ A | $D_f$ |
| Gaussian noise, $\sigma = 1$ | $16 \times 16 \times$ A | $D_f$ |
| Conv. $3 \times 3 \times$ B, stride 1, pad 1 | $16 \times 16 \times$ B | $D_f$ |
| Conv. $3 \times 3 \times$ B, stride 1, pad 1 | $16 \times 16 \times$ B | $D_f$ |
| Conv. $3 \times 3 \times$ B, stride 1, pad 1 | $16 \times 16 \times$ B | $D_f$ |
| Max-pooling $2 \times 2$, stride 2 | $8 \times 8 \times$ B | $D_f$ |
| Dropout ,rate=0.5 | $8 \times 8 \times$ B | $D_f$ |
| Gaussian noise, $\sigma = 1$ | $8 \times 8 \times$ B | $D_f$ |
| Conv. $3 \times 3 \times$ C, stride 1, pad 1 | $8 \times 8 \times$ C | $D_y$ |
| Conv. $3 \times 3 \times$ C, stride 1, pad 1 | $8 \times 8 \times$ C | $D_y$ |
| Conv. $3 \times 3 \times$ C, stride 1, pad 1 | $8 \times 8 \times$ C | $D_y$ |
| Global pooling | $1 \times 1 \times$ C | $D_y$ |
| Dense-softmax | #. of class | $D_y$ |

Table 1: CNN architecture of feature extractor $D_f$ and label predictor $D_y$.

| Layer type | Output size | model indice |
|---|---|---|
| Output of $D_f$ | $8 \times 8 \times$ B | $D_f$ |
| Reshape | $1 \times 64 \times$ B | $D_a$ |
| Dense-relu, 100 units | $1 \times 100$ | $D_a$ |
| Dense-sigmoid, 1 unit | 1 | $D_a$ |

Table 2: CNN architecture of domain discriminator $D_a$.

| Layer type | Output size | model indice |
|---|---|---|
| Input noise z | $1 \times 100$ | $G$ |
| Dense, 8192 units | $1 \times 8192$ | $G$ |
| Reshape | $4 \times 4 \times 512$ | $G$ |
| Transposed Conv. $5 \times 5 \times 256$, stride 2 | $8 \times 8 \times 256$ | $G$ |
| Transposed Conv. $5 \times 5 \times 128$, stride 2 | $16 \times 16 \times 128$ | $G$ |
| Transposed Conv.-tanh $5 \times 5 \times 3$, stride 2 | $32 \times 32 \times 3$ | $G$ |

Table 3: CNN architecture of generator $G$.

## 2 Hyper-parameters

Table 4 shows the hyper-parameters of each task.

| Task. | $\lambda_a$ | $\lambda_u$ |
|---|---|---|
| SVHN → MNIST | $10^{-2}$ | 0.5 |
| MNIST → SVHN | $10^{-2}$ | 0.1 |
| Syn-Num → SVHN | $10^{-2}$ | 1.0 |
| MNIST → MNIST-M | $10^{-2}$ | 0.5 |
| CIFAR → STL | $10^{-2}$ | 0.1 |
| STL → CIFAR | $10^{-2}$ | 0.1 |
| Syn-Signs → GTSRB | 0 | 0.5 |

Table 4: Hyper-parameters on seven UDA tasks.

## 3 Visualization

## References

[1] R. Shu, H. H. Bui, H. Narui, and S. Ermon. A DIRT-T approach to unsupervised domain adaptation. In *Proc. International Conference on Learning Representation*, 2018.

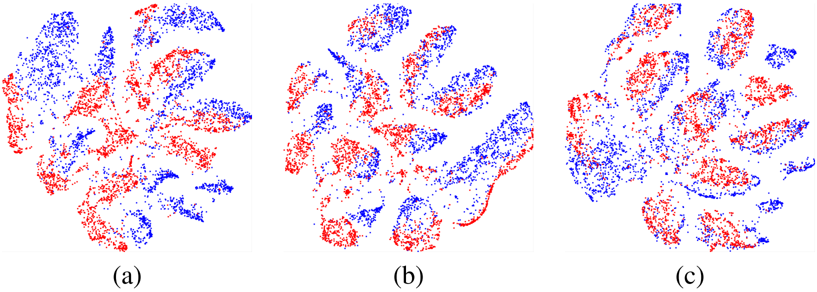
(a) (b) (c)

Figure 1: Visualization of feature representation from SVHN (blue) → MNIST (red) task via t-SNE. (a): DNN-SO; (b): DANN-O; (c): GAGL.

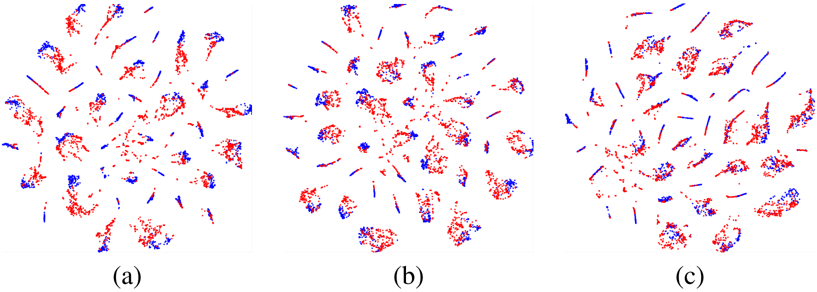
(a) (b) (c)

Figure 2: Visualization of feature representation from Syn-Signs (blue) → GTSRB (red) task via t-SNE. (a): DNN-SO; (b): DANN-O; (c): GAGL.

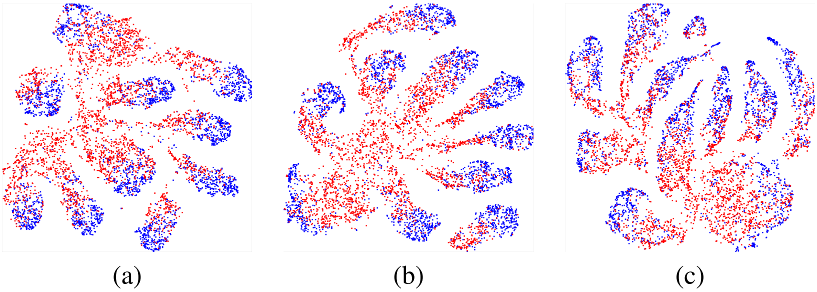
(a) (b) (c)

Figure 3: Visualization of feature representation from Syn-Num (blue) → SVHN (red) task via t-SNE. (a): DNN-SO; (b): DANN-O; (c): GAGL.