Supplementary Material

A. Additional Generated Images

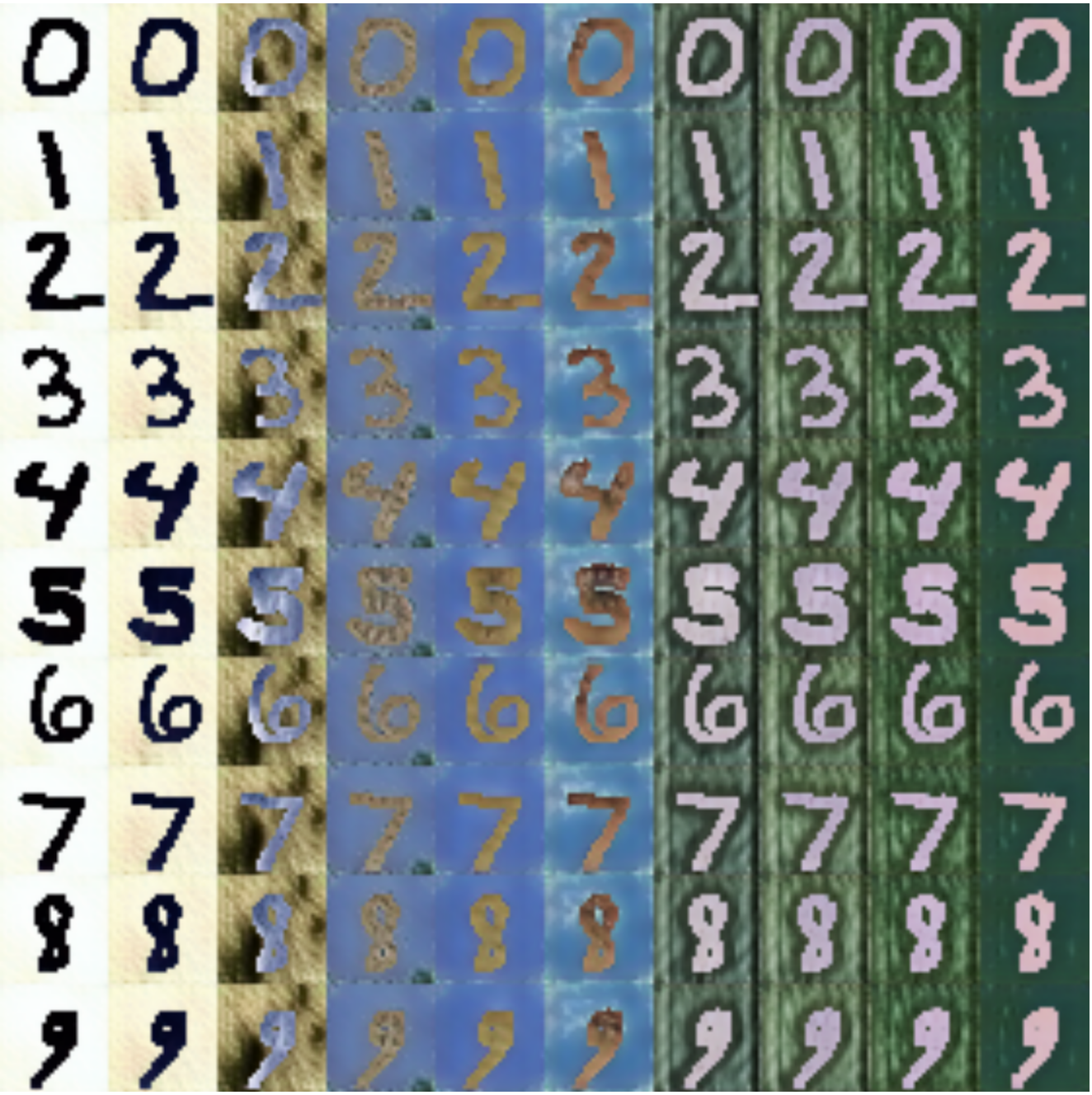

Figure 1. Linear interpolation between two random noise vectors demonstrates that the model is able to separate out style from content in the MNIST-M dataset. Each row is generated from the same MNIST digit, and each column is generated with the same noise vector.

B. Model Architectures and Parameters

We present the exact model architectures used for each experiment along with hyperparameters needed to reproduce results. The general form for the generator, G, and discriminator, D, are depicted in Figure 2 of the paper. For G, we vary the number of filters and the number of residual blocks. For D, we vary the amount of regularization and the number of layers.

Optimization consists of alternating optimization of the discriminator and task classifier parameters, referred to as the D step, with optimization of the generator parameters, referred to as the G step.

Unless otherwise specified, the following hyperparameters apply to all experiments:

- Batch size 32
- Learning rate decayed by 0.95 every 20,000 steps
- All convolutions have a 3x3 filter kernel
- Inject noise drawn from a zero centered Gaussian with stddev 0.2 after every layer of discriminator
- Dropout every layer in discriminator with keep probability of 90%
- Input noise vector is 10 dimensional sampled from a uniform distribution $\mathcal{U}(-1, 1)$
- We follow conventions from the DCGAN paper [36] for several aspects
  - An L2 weight decay of $1e^{-5}$ is applied to all parameters
  - Leaky ReLUs have a leakiness parameter of 0.2
  - Parameters initialized from zero centered Gaussian with stddev 0.02
  - We use the ADAM optimizer with $\beta_1 = 0.5$

B.1 USPS Experiments

The Generator and Discriminator are identical to the MNIST-M experiments.
Loss weights:

- Base learning rate is $2e^{-4}$
- The discriminator loss weight is 1.0
- The generator loss weight is 1.0
- The task classifier loss weight in G step is 1.0
- There is no similarity loss between the synthetic and generated images

B.2 MNIST-M Experiments (Paper Table 1)

Generator: The generator has 6 residual blocks with 64 filters each
Discriminator: The discriminator has 4 convolutions with 64, 128, 256, and 512 filters respectively. It has the same overall structure as paper Figure 2
Loss weights:

- Base learning rate is $1e^{-3}$
- The discriminator loss weight is 0.13
- The generator loss weight is 0.011
- The task classifier loss weight in G step is 0.01
- There is no similarity loss between the synthetic and generated images

B.3 LineMod Experiments

All experiments are run on a cluster of 10 TensorFlow workers. We benchmarked the inference time for the domain transfer on a single K80 GPU as 30 ms for a single example (averaged over 1000 runs) for the LineMod dataset.
Generator: The generator has 4 residual blocks with 64 filters each
Discriminator: The discriminator matches the depiction in paper Figure 2. The dropout keep probability is set to 35%.

Parameters without masked loss (Paper Table 2):

- Base learning rate is $2.2e^{-4}$, decayed by 0.75 every 95,000 steps
- The discriminator loss weight is 0.004
- The generator loss weight is 0.011
- The task classification loss weight is 1.0
- The task pose loss weight is 0.2
- The task classifier loss weight in G step is 0
- The task classifier is not trained on synthetic images
- There is no similarity loss between the synthetic and generated images

Parameters with masked loss (Paper Table 5):

- Base learning rate is $2.6e^{-4}$, decayed by 0.75 every 95,000 steps.
- The discriminator loss weight is 0.0088
- The generator loss weight is 0.011
- The task classification loss weight is 1.0
- The task pose loss weight is 0.29
- The task classifier loss weight in G step is 0
- The task classifier is not trained on synthetic images
- The MPSE loss weight is 22.9

C. InfoGAN Connection

In the case that T is a classifier, we can show that optimizing the task loss in the way described in the main text amounts to a variational approach to maximizing mutual information [2], akin to the InfoGAN model [7], between the predicted class and both the generated and the equivalent source images. The classification loss could be re-written, using conventions from [7] as:

$$\begin{aligned} \mathcal{L}_t &= -\mathbb{E}_{\mathbf{x}^s \sim \mathcal{D}^s}[\mathbb{E}_{y' \sim p(y|\mathbf{x}^s)} \log q(y'|\mathbf{x}^s)] \\ &\quad - \mathbb{E}_{\mathbf{x}^f \sim G(\mathbf{x}^s, \mathbf{z})}[\mathbb{E}_{y' \sim p(y|\mathbf{x}^f)} \log q(y'|\mathbf{x}^f)] \quad (7) \\ &\geq -I(y', \mathbf{x}^s) - I(y', \mathbf{x}^f) + 2H(y), \quad (8) \end{aligned}$$

where I represents mutual information, H represents entropy, $H(y)$ is assumed to be constant as in [7], y' is the random variable representing the class, and $q(y'|.)$ is an approximation of the posterior distribution $p(y'|.)$ and is expressed in our model with the classifier T. Again, notice that we maximize the mutual information of y' and the equivalent source and generated samples. By doing so, we are effectively regularizing the adaptation process to produce images that look similar for each class to the classifier T. This helps maintain the original content of the source image and avoids, for example, transforming all objects belonging to one class to look like objects belonging to another.

D. Deep Reconstruction-Classification Networks

Ghifary *et al*. [17] report a result of 91.80% accuracy on the MNIST $\rightarrow$ USPS domain pair, versus our result of 95.9%. We attempted to reproduce these results using their published code and our own implementation, but we were unable to achieve comparable performance.

Figure 2. Additional generation examples for the LineMod dataset. The left 4 columns are generated images and depth channels, while the corresponding right 4 columns are L2 nearest neighbors.

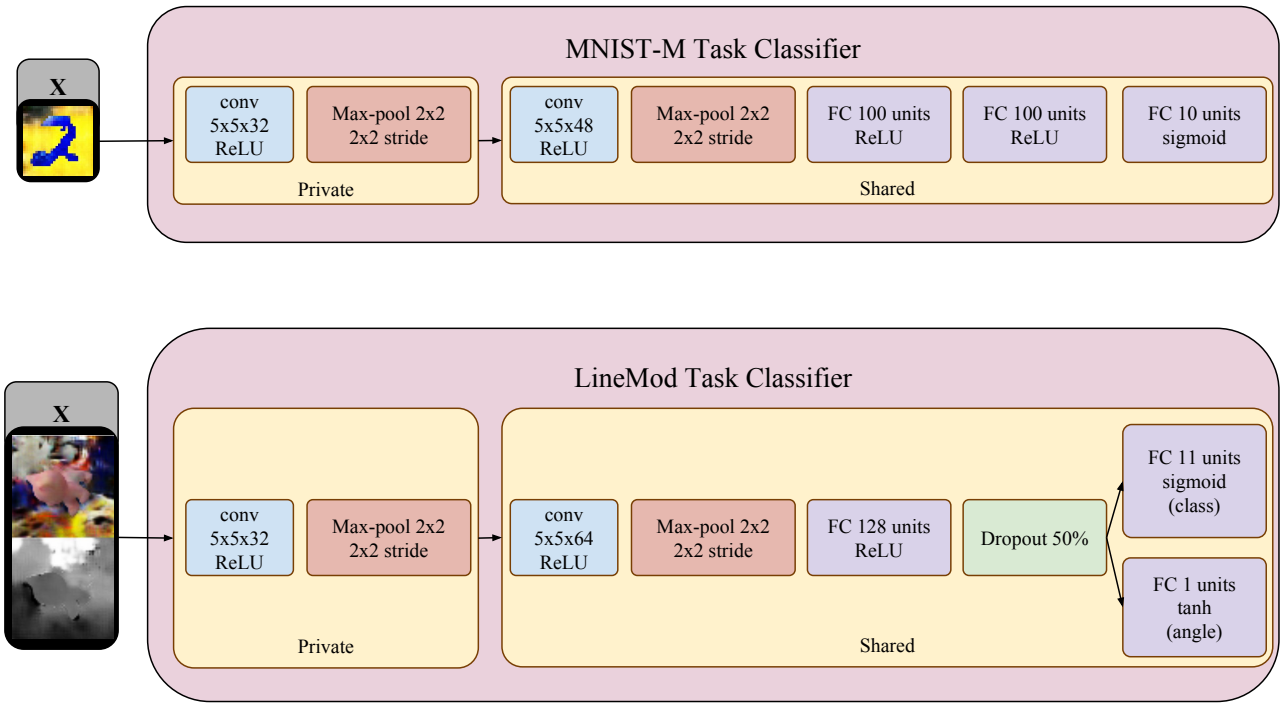

Figure 3. Task classifier (T) architectures for each dataset. We use the same task classifiers as [5, 14] to enable fair comparisons. The MNIST-M classifier is used for USPS, MNIST, and MNIST-M classification. During training, the task classifier is applied to both the synthetic and generated images when L_t^{source} is enabled (see Paper Table 5). The 'Private' parameters are only trained on one of these sets of images, while the 'Shared' parameters are shared between the two. At test time on the target domain images, the classifier is composed of the Shared parameters and the Private parameters which are part of the generated images classifier. These first private layers allow the classifier to share high level layers even when the synthetic and generated images have different channels (such as 1 channel MNIST and RGB MNIST-M).