StEP: Style-based Encoder Pre-training for Multi-modal Image Synthesis - Supplementary material

Moustafa Meshry Yixuan Ren Larry S. Davis Abhinav Shrivastava
University of Maryland, College Park

A. Appendix

A.1. Implementation details

Triplet selection requires computing the set of nearest and furthest neighbors to each anchor image. When pre-training using a large dataset, we found it sufficient to randomly sample a subset of 8000 images and sample triplets from this subset. This number was chosen to ensure fast nearest-neighbor computation.

The generator network G has a symmetric encoder-decoder architecture based on [1], with extra skip connections by concatenating feature maps of the encoder and decoder. We use a multiscale-patchGAN discriminator [1] with 3 scales and employ a LSGAN [2] loss. The mapper network $\mathcal{M}$ is a multi-layer perceptron (MLP) with three 128-dimensional hidden layers and a *tanh* activation function. For the reconstruction loss, we use the perceptual loss [3] evaluated at $\texttt{conv}_{i,2}$ for $i \in [1, 5]$ of VGG [4] with linear weights of $w_i = 1/2^{6-i}$ for $i \in [1, 5]$. The architecture of the style encoder E is adopted from [5], and we use a latent style vector $z \in \mathbb{R}^8$. Our optimizers setup is similar to that in [6]. We use three *Adam* optimizers: one for the generator G and encoder E, another for the discriminator D, and another optimizer for the generator G alone with $\beta_1 = 0, \beta_2 = 0.99$ for the three optimizers, and learning rates of $0.001, 0.001$ and 0.0001 respectively. We use a separate *Adam* optimizer for the mapper network $\mathcal{M}$ with $\beta_1 = 0.5, \beta_2 = 0.99$, and a learning rate of 0.01 with a decay rate of 0.7 applied every 50 steps. Relative weights for the loss terms are $\lambda_{\text{cGAN}} = 1$, $\lambda_{\text{rec}} = 0.02$ and $\lambda_{L2} = 0.01$ for the GAN loss, reconstruction loss, and $L2$ latent vector regularization respectively. When sampling triplets for any anchor image I_c, we use $k_c = 5, k_f = 13$ for the size of the set of close and far neighbors respectively.

A.2. Training time

Simplifying the training objective allows for faster training, as well as a larger batch size due to lower memory usage. Table 1 shows the processing time per 1000 training images for the baselines as well as different variations of our approach as defined in Table 2 in the main text.

Table 1: Training time (in seconds) per 1000 images for the baselines, as well as different versions of our approach (defined in Table 2 in the main text).

Approach	Batch size	time/kimg↓ (sec)	Max batch size	time/kimg↓ (sec)
Bicycle v1	8	93.11	12	85.36
MUNIT-p	8	155.72	8	155.72
Ours v1	8	145.50	8	145.50
Ours v2	8	98.55	12	93.04
Ours v3	8	64.92	16	53.80

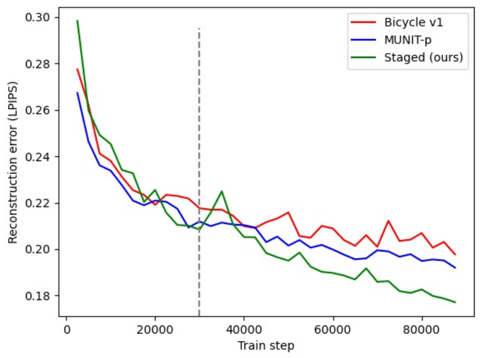


Figure 1: Convergence comparison between the proposed staged training (ours - v3) and the BicycleGAN baselines measured by the reconstruction error (LPIPS) of the validation set of the edges2handbags dataset. Dotted line shows the transition between stages 2 and 3 of our training (i.e, switching from a fixed E to finetuning both G and E together).

A.3. Convergence analysis

Figure 1 compares the convergence of our staged training compared to the BicycleGAN baselines. The dotted line in the graph marks the transition between stages 2 and 3 of our training (i.e, switching from a fixed pre-trained encoder E to finetuning both G and E together). We measure

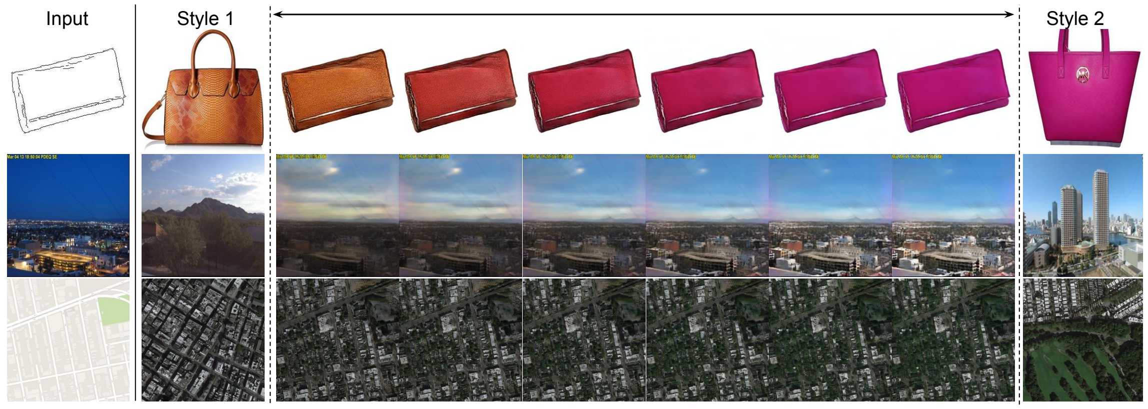


Figure 2: Style interpolation. Left column is the input to the generator G, second and last columns are input style images to the style encoder, and middle images are linear interpolation in the embedding space (figure better seen in zoom).

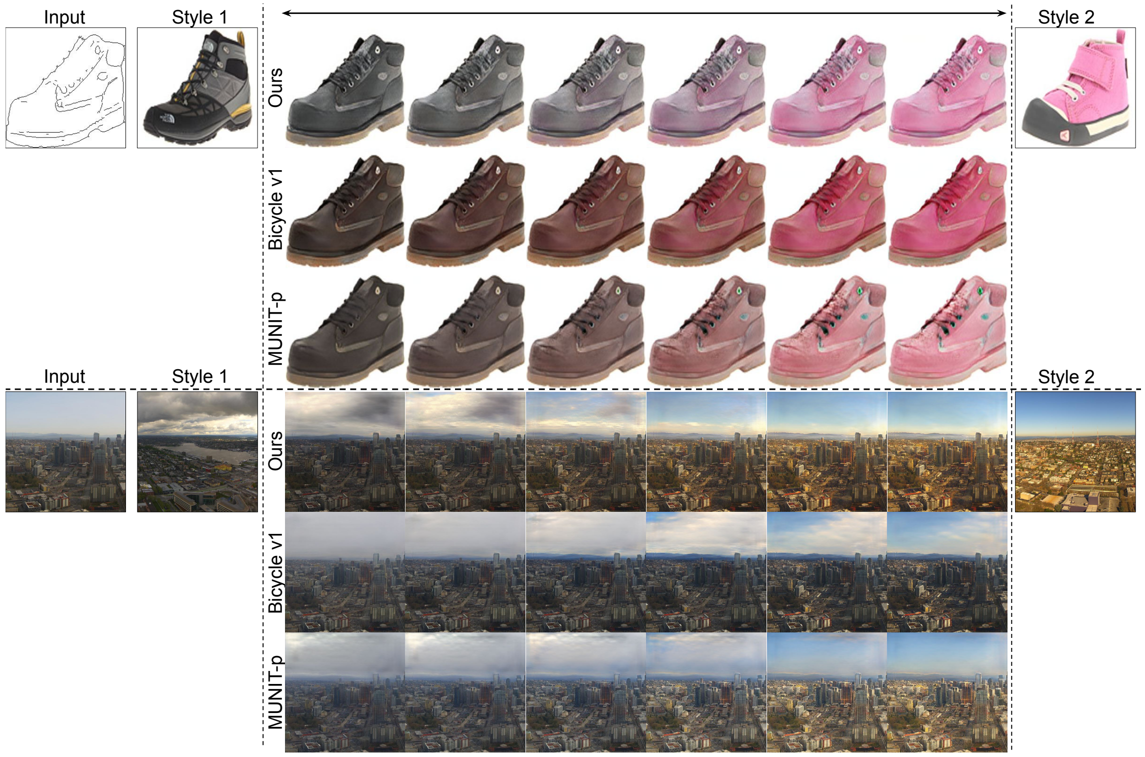


Figure 3: Style interpolation. Left column is the input to the generator G, second and last columns are input style images to the style encoder, and middle images are linear interpolation in the embedding space (figure better seen in zoom).

the reconstruction error (LPIPS) of the validation set of the edges2handbags dataset as the training progresses. Results show that with a fixed pre-trained encoder, our staged training starts with higher error than the baselines, but quickly drops to show similar performance as the baselines, and even beats the baselines before switching to stage 3 (marked by a dotted line). When starting to finetune the encoder E, we get a spike in the reconstruction error as the network adapts to the shift in the pre-trained embeddings, but then our staged training steadily widens the performance gap with the baselines. This shows the importance of the finetuning stage to tweak the pre-trained embeddings to better serve the image synthesis task for the target domain.

A.4. More quantitative comparison

We report the Inception Score (IS) computed over the validation set of various datasets in Table 2. Surprisingly, results after finetuning ("ours - stage 3") are slightly worse than those before finetuning ("ours - stage 2"), but both are still better than the baselines except for the maps dataset. We

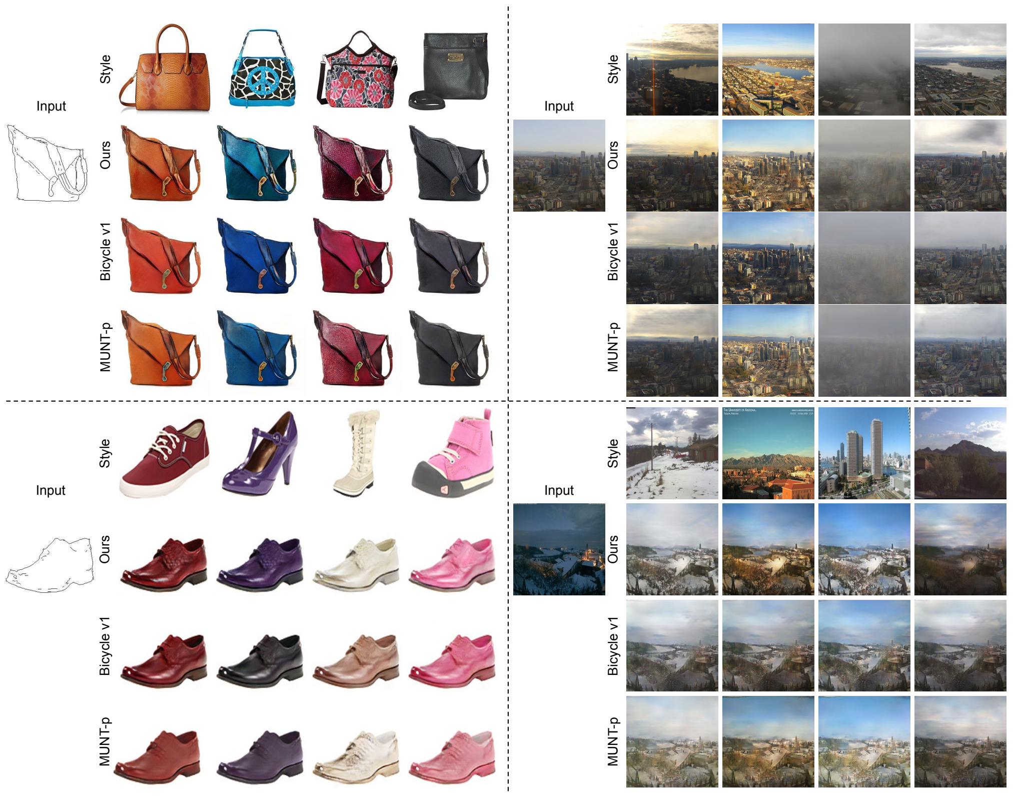


Figure 4: Style transfer comparison on different datasets. For each dataset, we apply different styles to the same input image and show the output of different methods.

also note that Inception Score is not very suited to image-to-image translation tasks, since it prefers output diversity with respect to ImageNet classes, not within-class diversity as in our case.

Table 2: Inception score comparison (higher is better) for different datasets.

	handbags	shoes	facades	night2day	maps	space needle
Bicycle v1	2.13	2.83	1.41	1.65	3.26	1.82
MUNIT-p	2.07	2.64	1.45	1.74	**3.57**	1.77
Ours - stage 2	**2.22**	2.75	**1.61**	1.76	3.32	**1.90**
Ours - stage 3	2.15	**2.85**	1.56	**1.84**	3.28	1.89

A.5. More style interpolations

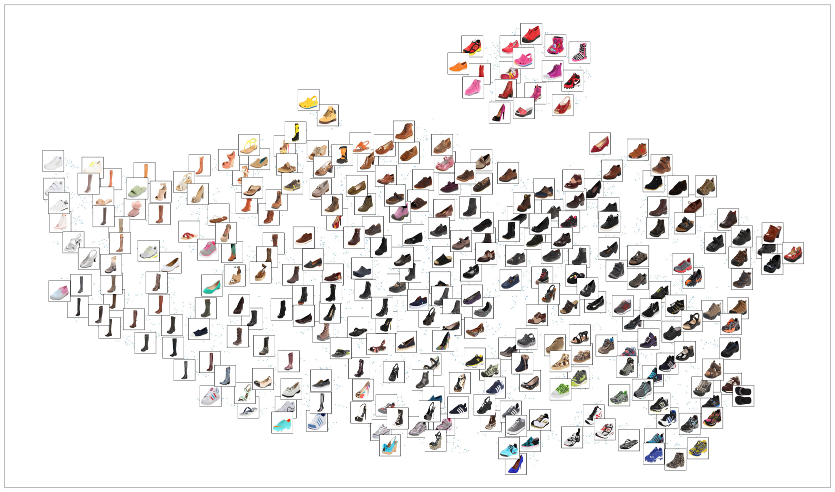
Figure 2 shows style interpolation on more datasets. Notice that, in the edges2handbags results, not only the color is transferred, but also the texture varies from non-smooth to smooth leather. Also, in the maps dataset, the density of bushes varies smoothly. Figure 3 further compares our interpolation results with the baselines. Our results show more complex interpolations, as evidenced by the change in lighting and cloud patterns, as well as more faithful style transfer compared to the baselines.

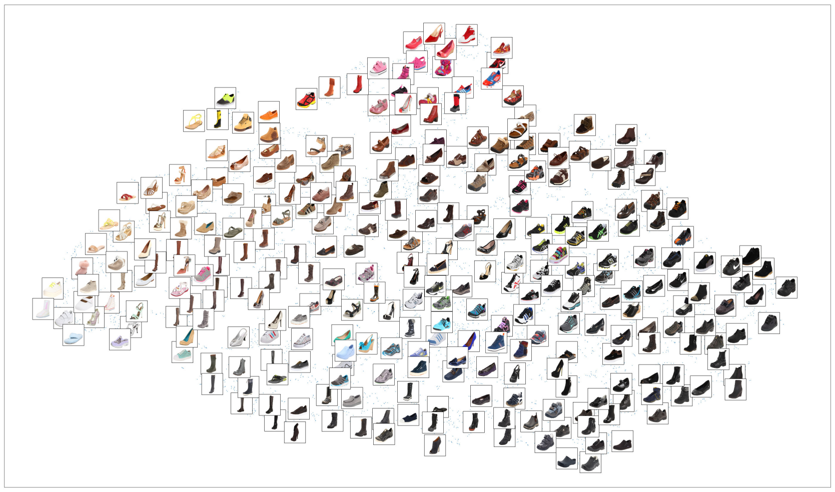
A.6. Style transfer comparison

We compare style transfer performance of our approach against that of the baselines in Figure 4. Our approach faithfully captures and transfers colors and weather conditions (including sky and surface lighting) compared to the baselines. We attribute the inferior results of the baselines to the reliance on VAEs to train the latent space. This is because noise added by VAEs means that slight changes to one style would still be mapped to the same point in the latent space, which limits the capacity of low dimensional latent space. On the other hand, our pre-trained embeddings don't rely on VAEs and hence, can discriminate between more styles.

A.7. Latent space visualization

Figure 5a visualizes the latent space learned by the style encoder E after pretraining and before finetuning (a), after finetuning (b), and the latent space learned by BicycleGAN [6] (c). The embedding learned through pre-training (i.e. before training the generator G) shows meaningful clusters, which verifies the validity of the proposed style-based pre-training. Finetuning smooths the style clusters and brings the latent space closer to that of BicycleGAN.


(a) Our approach: after style pretraining.


(b) Our approach: after finetuning.

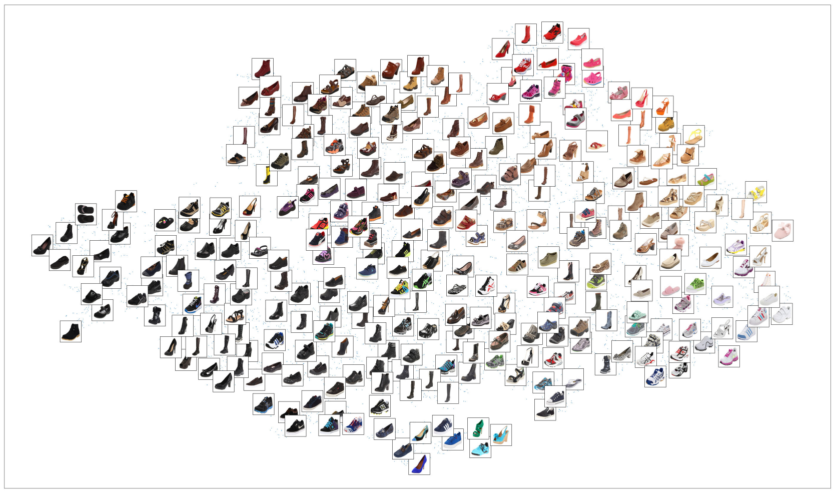

(c) BicycleGAN v1 baseline.

Figure 5: t-SNE plots for the latent style space learned by the style encoder E (a) after style pretraining, (b) after finetuning, and (c) using the BicycleGAN v1 baseline.

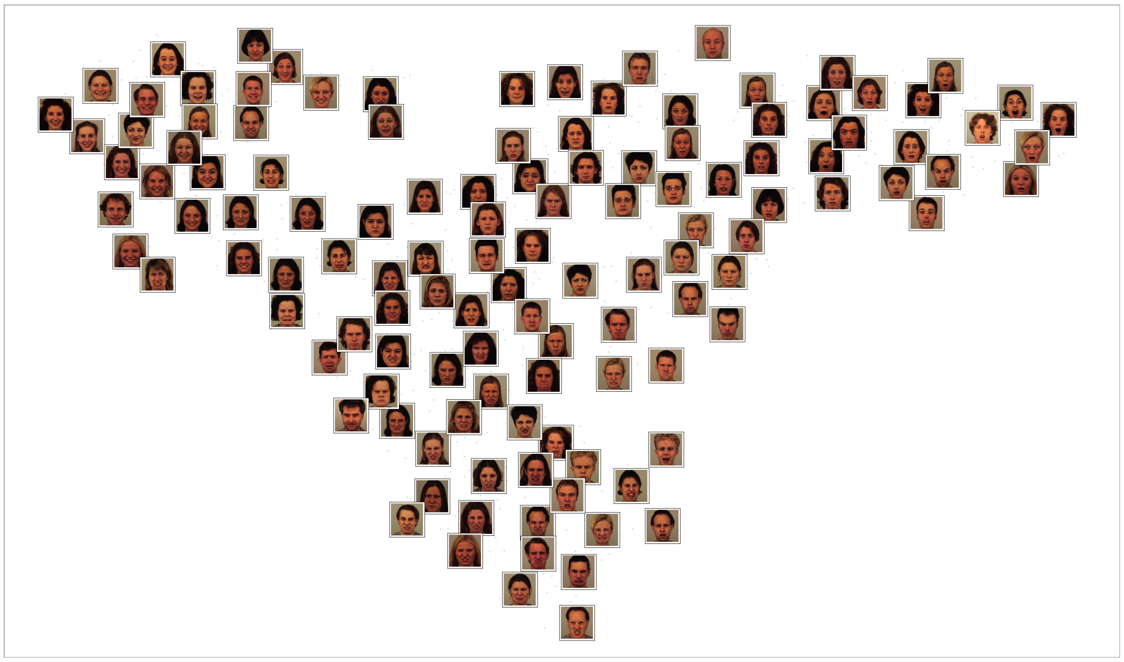


Figure 6: t-SNE plot for the pre-trained latent space learned for facial expressions on a subset of the KDEF dataset.

A.8. Encoder pre-training with non-style metrics

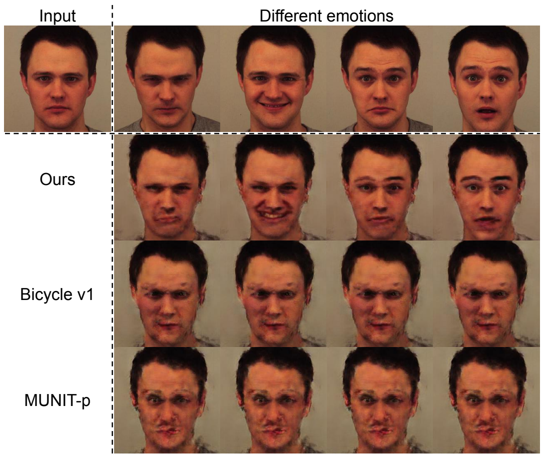


Figure 7: Emotion translation results. First row shows the input image, as well as the ground truth images from which we encode the latent emotion vector for reconstruction. Our staged training approach is able to achieve multi-modal synthesis, while the baselines collapse to a single mode.

Pre-training the encoder using a style-based triplet loss showed to be successful for multi-modal image translation tasks where the variability in the target domain is mainly color-based. This is shown in the results obtained on several benchmarks, even before the finetuning stage ("ours - stage 2" in Table 1 of the main text). We note though that the usage of style-loss as a distance metric for triplet sampling is just one choice and can be replaced with other distance metrics depending on the target application. Triplet sampling with style distance results in learning an embedding space where images with similar colors/styles lie closely in that space as shown in Section A.7. If, for example, we sample triplets instead based on the distance between VGG-Face [7] embeddings, the encoder will learn a latent space which is clustered by identity. In this section, we aim to validate that the proposed pre-training strategy can be extended to multi-modal image-to-image translation tasks with non-style variability. We inspect the task of manipulating facial expressions, where the input is a neutral face, and the output can have other emotions or facial expressions. For this task, similar emotions should be embedded closely in the latent space. We therefore use an off-the-shelf facial expression recognition system to compute the emotion similarity/distance between any pair of images. Specifically, we compute the emotion distance as the euclidean distance between the 512-dimensional feature map of the last layer of a pretrained classification network (e.g., [8]). We visualize the learned latent space in Figure 6, which shows clusters with similar emotions or facial expressions. We also show example translation results on a holdout set of the front-view images of the KDEF dataset [9] in Figure 7. We note that the generator successfully learns to manipulate facial expressions based solely on the pre-trained embeddings (without the finetuning stage). On the other hand, the BicycleGAN-based baselines collapsed to a single mode (over 3 different runs). This

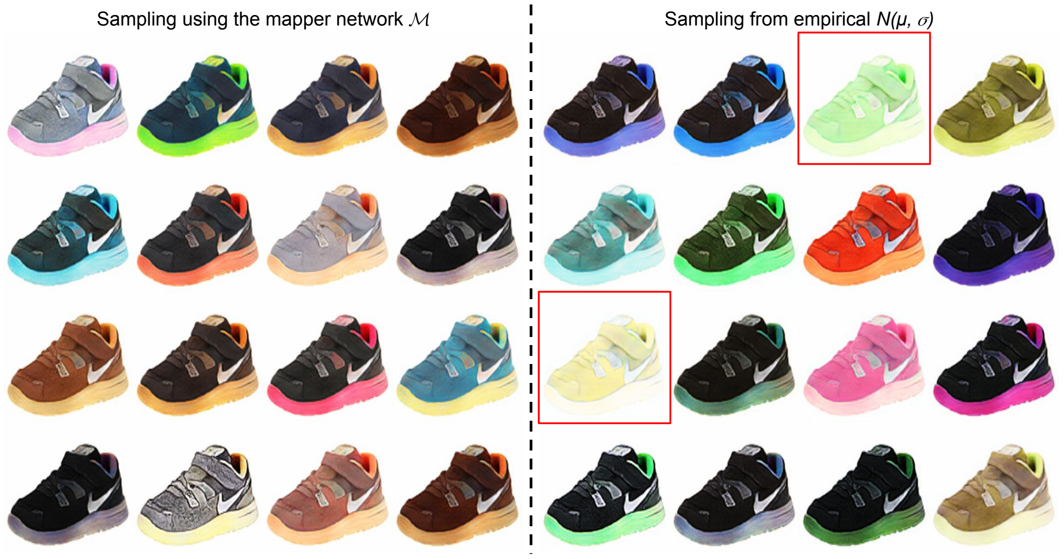

Sampling using the mapper network $\mathcal{M}$
Sampling from empirical $N(\mu, \sigma)$

Figure 8: Sixteen randomly sampled styles using both the mapper network $\mathcal{M}$ (left), as well as adhoc sampling from the empirically computed $N(\mu, \sigma)$ distribution of a $L2$-regularized latent space (right). Adhoc sampling could sample bad style codes outside the latent distribution (marked in red).

shows that our staged-training approach is stable and not sensitive to hyper-parameters, unlike the BicycleGAN baselines which will require careful hyper-parameter tuning to work properly on this task. We also point out that the poor output quality is mainly due to using a pixel-wise reconstruction loss for the generator training, while the input-output pairs in this dataset are not aligned. We didn't investigate improving the generator training as this is orthogonal to verifying the generalization of encoder pre-training.

A.9. Style sampling comparison

Figure 8 compares style sampling using the mapper network $\mathcal{M}$ vs adhoc sampling from the assumed $N(\mu, \sigma)$ of an $L2$-regularized latent space, where μ, σ are empirically computed from the training set. Note that adhoc sampling can sometimes sample bad style codes outside the distribution (e.g. third image in first row, and first image in third row in the right side of Figure 8), since the assumption that a $L2$-regularized space would yield normally distributed latents with zero mean and low standard deviation is not explicitly enforced.

References

[1] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 1

[2] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, 2017. 1

[3] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 1

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, 2014. 1

[5] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Kumar Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *ECCV*, 2018. 1

[6] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *NeurIPS*, 2017. 1, 3

[7] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *bmvc*, volume 1, page 6, 2015. 5

[8] Wu Jie. Facial expression recognition. `https://github.com/WuJie1010/Facial-Expression-Recognition.Pytorch`, 2018. Accessed: 2019-09-22. 5

[9] KDEF. Karolinska directed emotional faces (kdef) dataset. `http://kdef.se`, 2017. Accessed: 2019-09-22. 5