Supplementary Materials of HistoGAN: Controlling Colors of GAN-Generated and Real Images via Color Histograms

Mahmoud Afifi Marcus A. Brubaker Michael S. Brown

York University

{mafifi,mab,mbrown}@eecs.yorku.ca

This supplementary materials provide additional details and results of our method. We first discuss additional details of our networks in Sec. 1. Then, we explain the training details in Sec. 2. Afterwards, Sec. 3 presents ablation experiments carried out to validate our choice of hyperparameters and loss terms. In Sec. 4, we present our experiments performed to train a "universal" recoloring model to recolor images taken from arbitrary domains. Sec. 5 discusses failure cases of our method. Such failure cases can often be mitigated by applying simple post-processing. The post-processing details in Sec. 6. Sec. 6 also discuss post-processing to deal with high-resolution images. Lastly, additional results are given in Sec. 7.

1. Details of Our Networks

Our discriminator network, used in all of our experiments, consists of a sequence of $\log_2(N)1$ residual blocks, where N is the image width/height, and the last layer is an fully connected (fc) layer that produces a scalar feature. The first block accepts a three-channel input image and produce m output channels. Then, each block i produces $2m_{i-1}$ output channels (i.e., duplicate the number of output channels of the previous block). The details of the residual blocks used to build our discriminator network are shown in Fig. 1.

Figure 2 provides the details of our encoder, decoder and GAN blocks used in our ReHistoGAN (used for image recoloring). As shown, we modified the last two blocks of our HistoGAN's to accept the latent feature passed from the first two blocks of our encoder. This modification helps our HistoGAN's head to consider both information of the input image structure and the target histogram in the recoloring process.

2. Training Details

We train our networks using an NVIDIA TITAN X (Pascal) GPU. For HistoGAN training, we optimized both the generator and discriminator networks using the diffGrad optimizer [11]. In all experiments, we set the histogram bin, h, to 64 and the fall-off parameter of our histogram's bins, τ, was set to 0.02. We adopted the exponential moving average of generator network's weights [15, 16] with the path length penalty, introduced in StyleGAN [16], every 32 iterations to train our generator network. Due to the hardware limitation, we used mini-batch of 2 with accumulated gradients every 16 iteration steps and we set the image's dimension, N, to 256. We set the scale factor of the Hellinger distance loss, α, to 2 (see Sec. 3 for an ablation study).

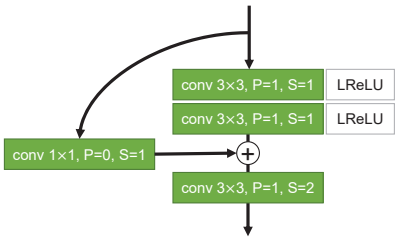

Figure 1: Details of the residual discriminator block used to reconstruct our discriminator network. The term P and S refer to the padding and stride used in each layer.

As mentioned in the main paper, we trained our HistoGAN using several domain datasets, including: human faces [15], flowers [26], cats [9], dogs [17], birds [32], anime faces [8], human hands [1], bedrooms [34], cars [18], and aerial scenes [24]. We further trained our HistoGAN using 4,316 landscape images collected from Flickr[1]. The collected images have one of the following copyright licenses: no known copyright restrictions, Public Domain Dedication (CC0), or Public Domain Mark. See Fig. 3 for representative examples from the landscape set.

To train our ReHistoGAN, we used the diffGrad optimizer [11] with the same mini-batch size used to train our HistoGAN. We trained our network using the following hyperparameters $\alpha = 2$, $\beta = 1.5$, $\gamma = 32$ for 100,000 iterations. Then, we continued training using $\alpha = 2$, $\beta = 1$, $\gamma = 8$ for additional 30,000 iterations to reduce potential

[1] Our landscape image set: https://github.com/mahmoudnafifi/HistoGAN

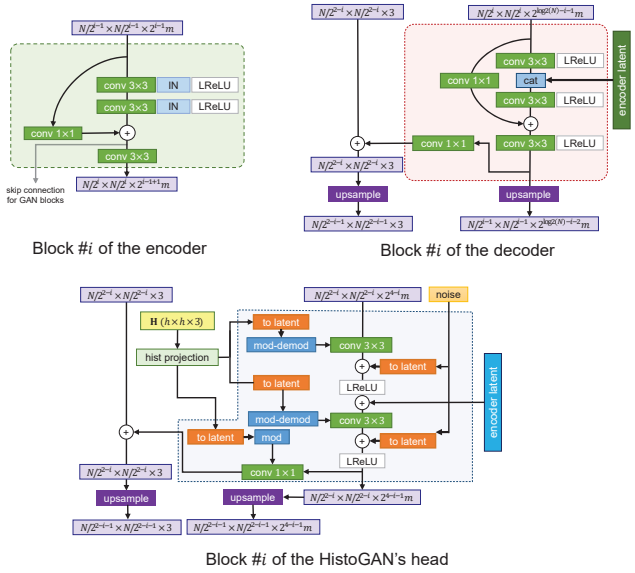

Figure 2: Details of our ReHistoGAN network. We modified the last two blocks of our HistoGAN by adding a gate for the processed skipped features from the first two blocks of our encoder.

Figure 3: Examples taken from our set of 4,316 landscape images collected from Flickr.

artifacts in recoloring (see Sec. 3 for an ablation study).

3. Ablation Studies

We carried out a set of ablation experiments to study the effect of different values of hyperparameters used in the main paper. Additionally, we show results obtained by variations in our loss terms.

We begin by studying the effect of the scale factor, α, used in the loss function to train our HistoGAN. This scale factor was used to control strength of the histogram loss

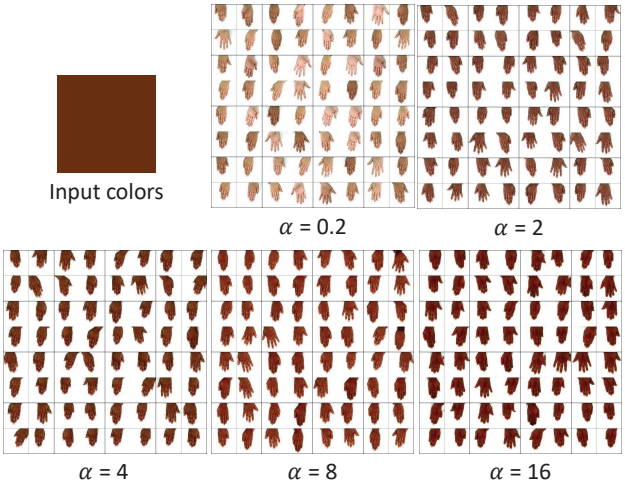

Figure 4: Results obtained by training our HistoGAN in hand images [1] using different values of α.

Table 1: Results of our HistoGAN using different values of α. In this set of experiments, we used the Hands dataset [1] as our target domain. The term FID stands for the Frechét inception distance metric [14]. The term KL Div. refers to the KL divergence between the histograms of the input image and generated image, while the term H. dis. refers to Hellinger distance.

α	FID	RGB-uv hist.	
		KL Div.	H dist.
0.2	1.9950	0.3935	0.3207
2	**2.2438**	**0.0533**	**0.1085**
4	6.8750	0.0408	0.0956
8	9.4101	0.0296	0.0822
16	15.747	0.0237	0.0743

term. In this set of experiments, we used the 11K Hands dataset [1] to be our target domain and trained our HistoGAN with the following values of α: 0.2, 2, 4, 8, and 16. Table 1 shows the evaluation results using the Frechét inception distance (FID) metric [14], the KL divergence, and Hellinger distance. The KL divergence and Hellinger distance were used to measure the similarity between the target histogram and the histogram of GAN-generated images. Qualitative comparisons are shown in Fig. 4

Figure 5 shows examples of recoloring results obtained by trained ReHistoGAN models using different combination values of α, β, γ. As can be seen, a lower value of the scale factor, α, of the histogram loss term results in ignoring our network to the target colors, while higher values of the scale factor, γ, of the discriminator loss term, make our method too fixated on producing realistic output images, regardless of achieving the recoloring (i.e., tending to re-produce the input image as is).

In the recoloring loss, we used a reconstruction loss term

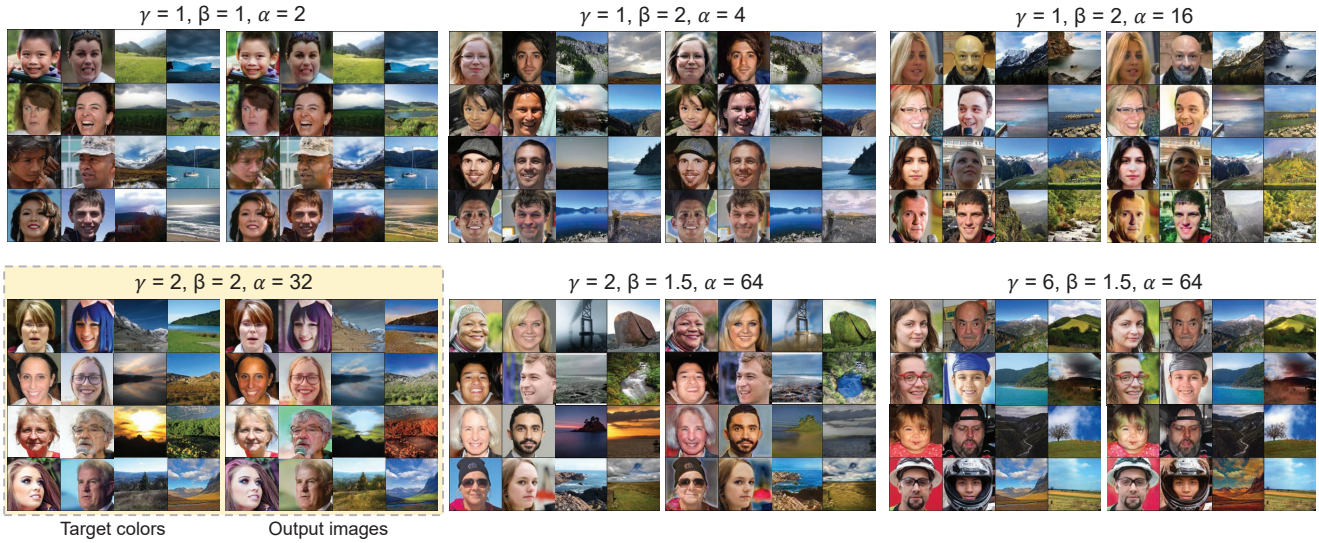

Figure 5: Results of recoloring by training our recoloring network using different values of α, β, and γ hyperparameters. The highlighted results refer to the settings used to produce the reported results in the main paper and the supplementary materials.

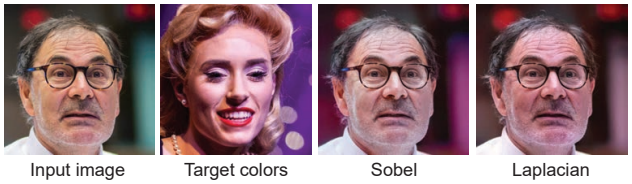

Figure 6: Results of two different kernels used to compute the reconstruction loss term.

to retain the input image's spatial details in the output recolored image. Inspired by image seamless cloning methods (e.g., [3, 27]), our reconstruction loss is based on the derivative of the input image. We have examined two different kernels, which are: the vertical and horizontal 3×3 Sobel kernels (i.e., the first-order derivative approximation) and the 3×3 Laplacian kernel (i.e., the second-order derivative). We found that training using both kernels give reasonably good results, while the Laplacian kernel produces more compiling results in most cases; see Fig. 6 for an example.

In the main paper, we introduced a variance loss term to encourage our network to avoid the global color cast solution for image recoloring. Figure 7 shows an example of the global color cast problem, where the network applies a global color shift to the input image to match the target histogram. As shown in Fig. 7 after training our network with the variance loss, this problem is reduced.

Figure 7: The impact of the variance loss term. The shown results were obtained by training our ReHistoGAN with and without the variance loss term.

4. Universal ReHistoGAN Model

As the case of most GAN methods, our ReHistoGAN targets a specific object domain to achieve the image recoloring task. This restriction may hinder the generalization of our method to deal with images taken from arbitrary domains. To deal with that, we collected images from a different domain, aiming to represent the "universal" object domain.

Specifically, our training set of images contains ~2.4 million images collected from different image datasets. These datasets are: collection from the Open Images dataset [19], the MIT-Adobe FiveK dataset [6], the Microsoft COCO dataset [22], the CelebA dataset [23], the Caltech-UCSD birds-200-2011 dataset [32], the Cats dataset [9], the Dogs dataset [17], the Cars dataset [18], the OxFord Flowers dataset [26], the LSUN dataset [34], the ADE20K dataset [35, 36], and the FFHQ dataset [15]. We also added Flickr images collected using the following keywords: `landscape`, `people`, `person`, `portrait`, `field`, `city`, `sunset`, `beach`, `animals`, `living room`, `home`, `house`, `night`, `street`, `desert`, `food`. We have excluded any grayscale image from the collected image set.

We trained our "universal" model using $m = 18$ on this

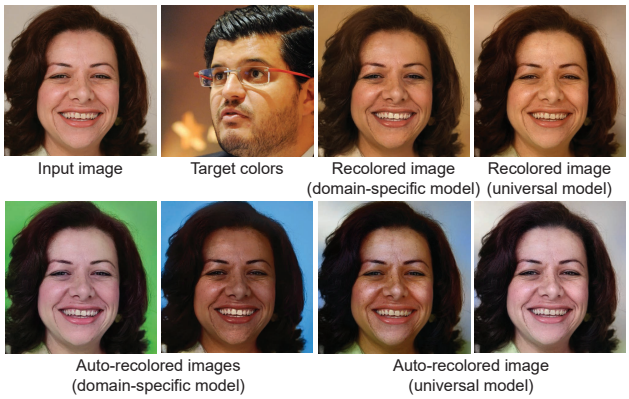

Figure 8: Results of domain-specific and universal ReHistoGAN models. We show results of using a given target histogram for recoloring and two examples of the auto recoloring results of each model.

collected set of 2,402,006 images from several domains. The diffGrad optimizer [11] was used to minimize the same generator loss described in the main paper using the following hyperparameters $\alpha = 2$, $\beta = 1.5$, $\gamma = 32$ for 150,000 iterations. Then, we used $\alpha = 2$, $\beta = 1$, $\gamma = 8$ to train the model for additional 350,000 iterations. We set the mini-batch size to 8 with an accumulated gradient every 24 iterations. Figure 8 show results of our domain-specific and universal models for image recoloring. As can be seen, both models produce realistic recoloring, though the universal model tends to produce recolored images with less vivid colors compared to our domain-specific model. Additional examples of auto recoloring using our universal model are shown in Fig. 9.

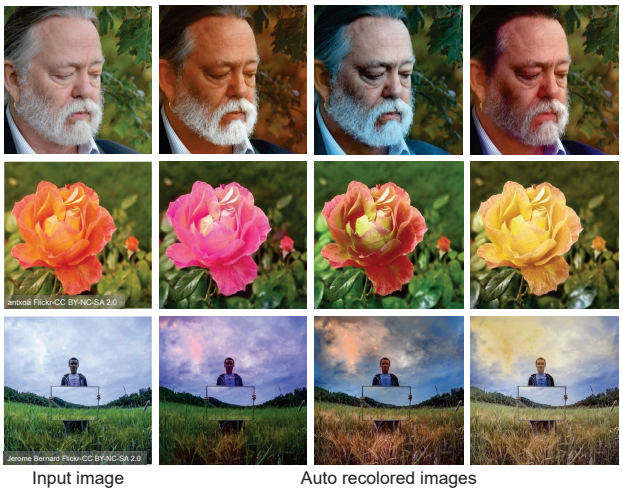

Figure 9: Auto recoloring using our universal ReHistoGAN model.

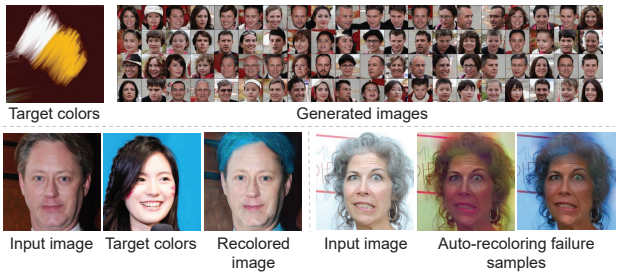

Figure 10: Failure cases of HistoGAN and ReHistoGAN. Our HistoGAN fails sometimes to consider all colors of target histogram in the generated image. Color bleeding is another problem that could occur in ReHistoGAN's results, where our network could not properly allocate the target (or sampled) histogram colors in the recolored image.

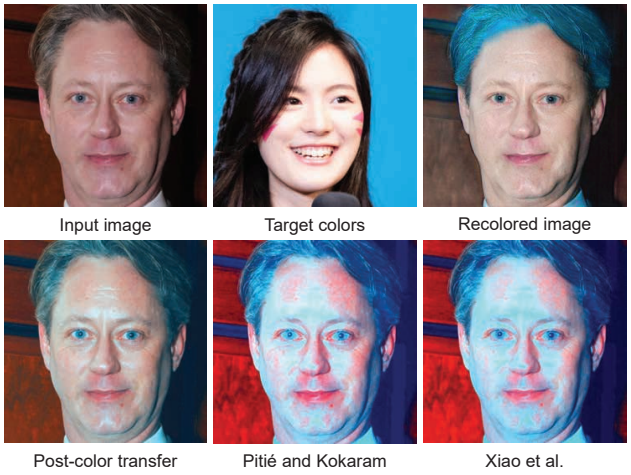

Figure 11: To reduce potential color bleeding artifacts, it is possible to apply a post-color transfer to our initial recolored image colors to the input image. The results of adopting this strategy are better than applying the color transfer to the input image in the first place. Here, we use the color transfer method proposed by Pitié and Kokaram [28] as our post-color transfer method. We also show the results of directly applying Pitié and Kokaram's [28] method to the input image.

5. Limitations

Our method fails in some cases, where the trained HistoGAN could not properly extract the target color information represented in the histogram feature. This problem is due to the inherent limitation of the 2D projected representation of the original target color distribution, where different colors are mapped to the same chromaticity value in the projected space. This is shown in Fig. 10-top, where the GAN-generated images do not have all colors in the given target histogram. Another failure case can occur in image

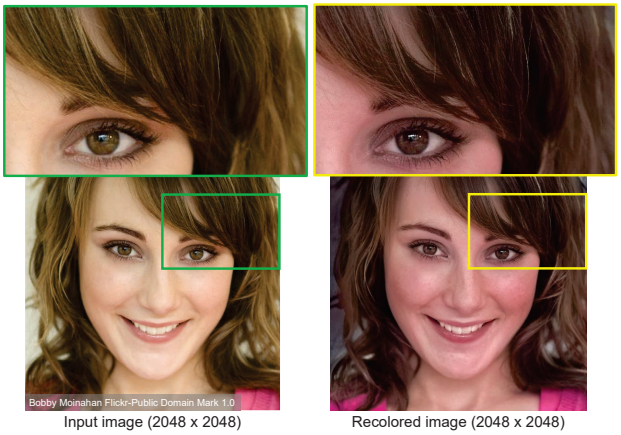

Figure 12: We apply the bilateral guided upsampling [7] as a post-processing to reduce potential artifacts of dealing with high-resolution images in the inference phase. In the shown example, we show our results of recoloring using an input image with 2048×2048 pixels.

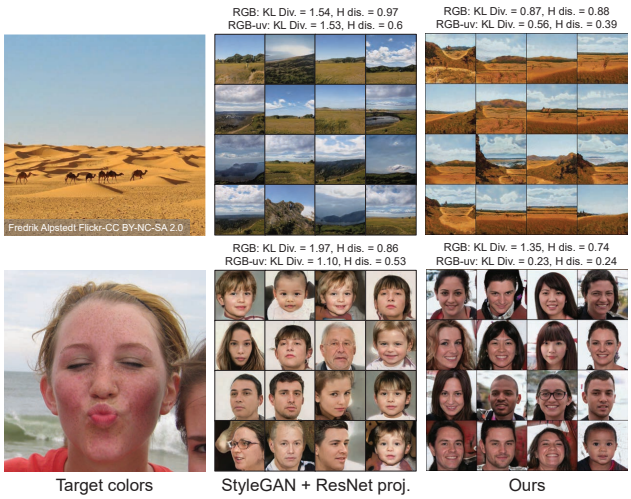

Figure 13: Comparison with generated images using StyleGAN [16] with latent space projection (see the main paper for more details) and our results.

recoloring, where the recolored images could have some color-bleeding artifacts due to errors in allocating the target/sampled histogram colors in the recolored image. This can be shown in Fig. 10-bottom

6. Post-Processing

As discussed in Sec. 5, our method produces, in some times, results with color bleeding, especially when the target histogram feature has unsuitable color distribution for the content of the input image. This color-bleeding problem can be mitigated using a post-process color transfer between the input image and our initial recoloring. Surprisingly, this post-processing mapping produces results better than adopting the mapping in the first place—namely, applying the color transfer mapping without having our intermediate recoloring result.

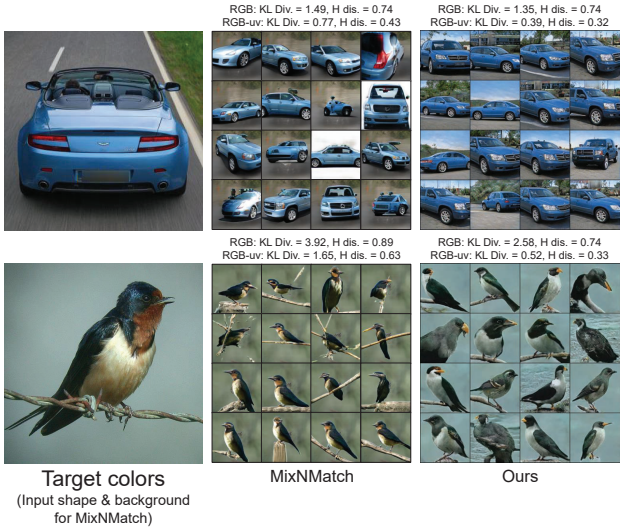

Figure 14: Additional comparison with the MixNMatch method [21]. In these examples, the target images were used as input shape and background images for the MixNMatch method.

Figure 11 shows an example of applying Pitié, and Kokaram's method [28] as a post-processing color transfer to map the colors of the input image to the colors of our recolored image. In the shown figure, we also show the result of using the same color transfer method – namely, Pitié and Kokaram's method [28] – to transfer the colors of the input image directly to the colors of the target image. As shown, the result of using our post-process strategy has a better perceptual quality.

Note that except for this figure (i.e., Fig. 11), we *did not* adopted this post-processing strategy to produce the reported results in the main paper or the supplementary materials. We discussed it here as a solution to reduce the potential color bleeding problem for completeness.

As our image-recoloring architecture is a fully convolutional network, we can process testing images in any arbitrary size. However, as we trained our models on a specific range of effective receptive fields (i.e., our input image size is 256), processing images with very high resolution may cause artifacts. To that end, we follow the post-processing approach used in [2] to deal with high-resolution images (e.g., 16-megapixel) without affecting the quality of the recolored image.

Specifically, we resize the input image to 256×256 pixels before processing it with our network. Afterward, we apply the bilateral guided upsampling [7] to construct the

Figure 15: Our HistoGAN can be used to generate "unlimited" number of random samples, exactly like traditional StyleGANs [15, 16], by sampling from a pre-defined set of histograms to generate target histograms. In the shown figure, we show generated images by StyleGAN [15, 16] and our HistoGAN. In each row of the StyleGAN-generated images, we fixed the fine-style vector of the last two blocks of the StyleGAN, as these blocks are shown to control the fine-style of the generated image [16]. We also fixed the generated histogram for each row of our HistoGAN-generated images.

mapping from the resized input image and our recoloring result. Then, we apply the constructed bilateral grid to the input image in its original dimensions. Figure 12 shows an example of our recoloring result for a high-resolution image (2048×2048 pixels). As can be seen, our result has the same resolution as the input image with no artifacts.

7. Additional Results

This section provides additional results generated by our HistoGAN and ReHistoGAN. As discussed in the main paper, we trained a regression ResNet [13] model to learn the back-projection from the generated images into the cor-

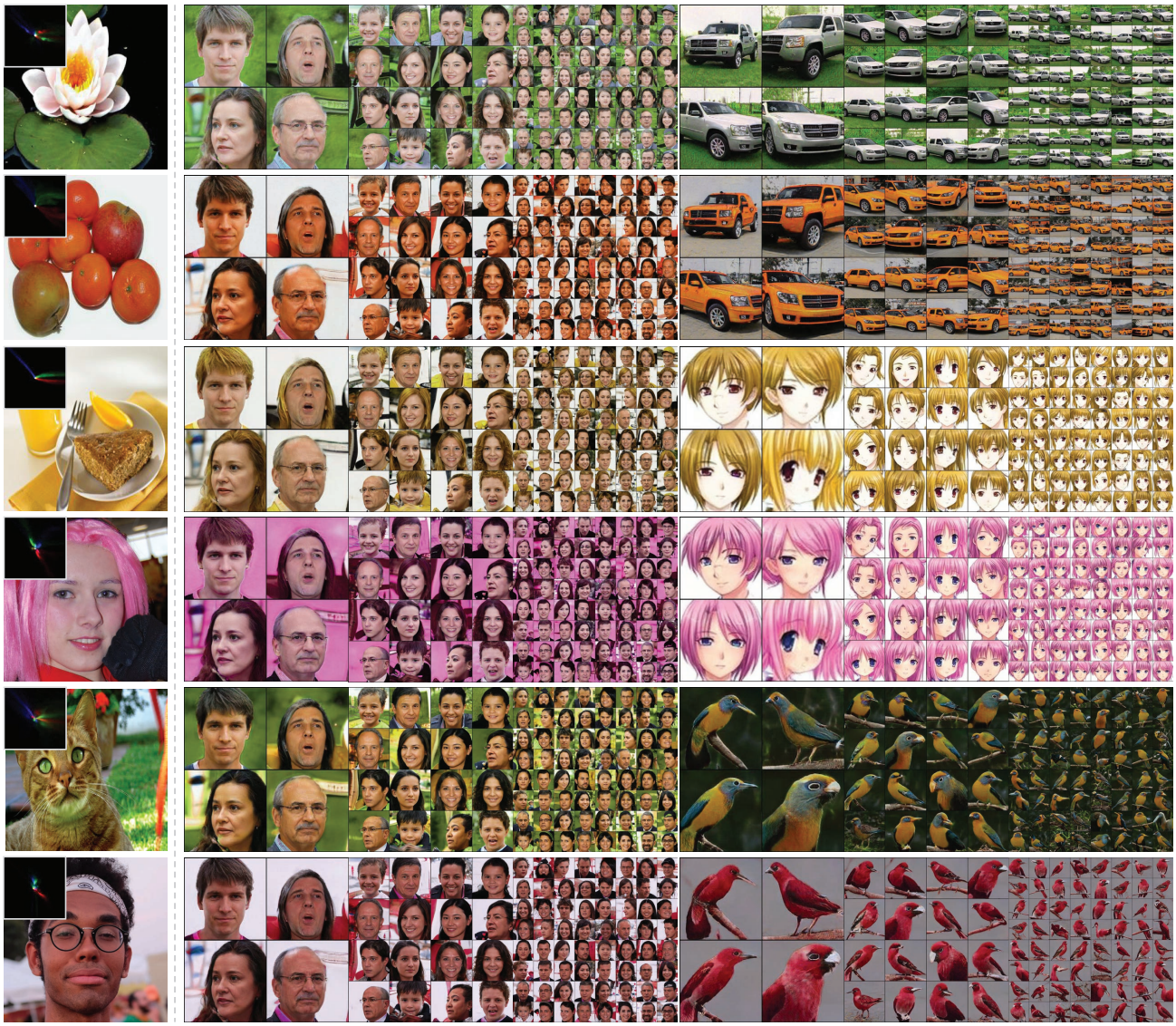

Figure 16: Additional examples of generated images using our HistoGAN. The target colors histograms were computed from the shown target images (left column).

responding fine-style vectors of StyleGAN [16]. This regression model was used to compareHistoGAN and StyleGAN's ability to control the generated images' colors given a target color distribution. Figure 13 shows a qualitative comparison between the results of our HistoGAN and StyleGAN with this projection approach. We show additional qualitative comparisons with the recent MixNMatch method [21] in Fig. 14. In the shown figures, we show the KL divergence and the Hellinger distance between the histograms of the GAN-generated images and the target histogram.

Our HistoGAN, along with the sampling procedure (used for auto recoloring in the main paper) can be used to turn our HistoGAN into a traditional GAN method, where there is no need for any user intervention to input the target histograms. Figure 15 shows an example of using our sampling procedure to generate random histogram samples. The generated histogram samples are used by HistoGAN to generate "unlimited" number of samples. In the shown figure, we compare between our HistoGAN results, using the generated histograms, with StyleGAN [16]. In Fig.15-(A), each row shows generated examples with a fixed fine-style vectors, which are used by the last two blocks of the StyleGAN as these blocks are shown to control the fine-style (e.g., colors, lighting, etc.) of the generated image [15, 16]. In Fig.15-(B), each row shows generated images using our

Figure 17: Additional comparisons with image recoloring/style transfer methods. We compare our results with results of the following methods: Reinhard et al., [29], Xiao et al., [33], Pitié and Kokaram [28], Nguyen et al., [25], and Sheng et al., [30].

HistoGAN with a fixed generated histogram. As shown in the figure, our HistoGAN generates samples with a higher color diversity compared to StyleGAN results.

Figure 16 shows additional HistoGAN-generated images from different domains. In each row, we show example images generated using the corresponding input target colors. We fixed the coarse- and middle-style vectors, for each domain, to show the response of our HistoGAN to changes in the target histograms.

In the main paper, we showed comparisons with different image recoloring and style transfer methods. Figure 17 shows additional qualitative comparisons. Note that Gaty et al.'s optimization method [12] takes ∼4 minutes to process a single image. In contrast, our ReHistoGAN processes a single image in ∼0.5 seconds without the guided upsampling procedure [7], and ∼22 seconds with an unoptimized implementation of the guided upsampling using a single GTX 1080 GPU. Further qualitative examples are shown in Fig. 18. As can be seen, our ReHistoGAN successfully transfers the target colors to the recolored images naturally.

As mentioned in the main paper, there are a few attempts to achieve auto recoloring (e.g., [4, 5, 10, 20]). The high-resolution daytime translation (HiDT) method [5], for example, achieves the auto-style transfer by sampling from a pre-defined set of target styles. We compared our method and the HiDT method in the main paper, where we used one of the pre-defined target styles as our target histogram. This idea of having a pre-defined set of target styles was originally proposed in [20], where a set of transient attributes are used to search in a dataset of different target styles. These methods, however, are restricted to the semantic content of the target styles to match the semantic content of training/testing input images. Unlike these auto recoloring/style transfer methods, our ReHistoGAN can deal with histograms taken from any arbitrary domain, as shown in our results in the main paper and these supplementary materials. In Fig. 19, we show qualitative comparisons of the recoloring results using our universal ReHistoGAN and the method proposed in [20].

Another strategy for image recoloring is to learn a diverse colorization model. That is, the input image is converted to grayscale, and then a trained method for diverse colorization can generate different colorized versions of the input image. In Fig. 20, we show a qualitative comparison with the diverse colorization method proposed by Deshpande et al., [10].

Lastly, we show additional qualitative comparisons with the recent auto-recoloring method proposed by Afifi et al., [4] in Fig. 21. The figure shows the results of domain-specific ReHistoGAN models (the first four rows) and the universal ReHistoGAN model (the last three rows). As can be seen from the shown figures, our ReHistoGAN arguably

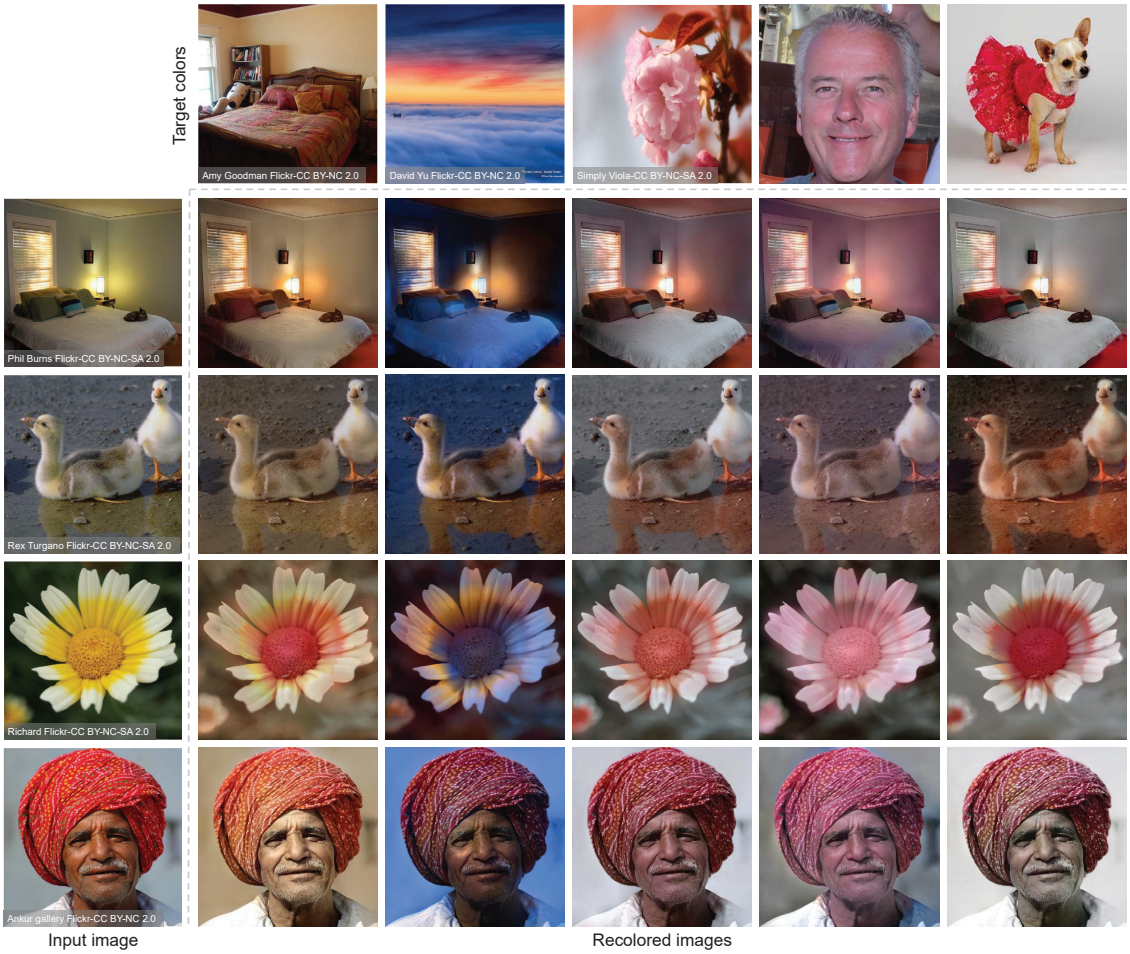

Figure 18: Additional results for image recoloring. We recolor input images, shown in the right by feeding our network with the target histograms of images shown in the top.

produces more realistic recoloring compared to the recoloring results produced by other auto recoloring methods.

References

[1] Mahmoud Afifi. 11K hands: Gender recognition and biometric identification using a large dataset of hand images. *Multimedia Tools and Applications*, 78(15):20835–20854, 2019. 1, 2

[2] Mahmoud Afifi, Konstantinos G Derpanis, Björn Ommer, and Michael S Brown. Learning multi-scale photo exposure correction. In *CVPR*, 2021. 5

[3] Mahmoud Afifi and Khaled F Hussain. MPB: A modified Poisson blending technique. *Computational Visual Media*, 1(4):331–341, 2015. 3

[4] Mahmoud Afifi, Brian L Price, Scott Cohen, and Michael S Brown. Image recoloring based on object color distributions. In *Eurographics 2019 (short papers)*, 2019. 8, 11

[5] Ivan Anokhin, Pavel Solovev, Denis Korzhenkov, Alexey Kharlamov, Taras Khakhulin, Aleksei Silvestrov, Sergey Nikolenko, Victor Lempitsky, and Gleb Sterkin. High-resolution daytime translation without domain labels. In *CVPR*, 2020. 8

[6] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. Learning photographic global tonal adjustment with a database of input / output image pairs. In *CVPR*, 2011. 3

[7] Jiawen Chen, Andrew Adams, Neal Wadhwa, and Samuel W Hasinoff. Bilateral guided upsampling. *ACM Transactions on Graphics (TOG)*, 35(6):1–8, 2016. 5, 8

[8] Spencer Churchill. Anime face dataset. https://www.kaggle.com/splcher/animefacedataset. [Online; accessed October 27, 2020]. 1

[9] Chris Crawford and NAIN. Cat dataset. https://www.kaggle.com/crawford/cat-dataset. [Online; accessed October 27, 2020]. 1, 3

[10] Aditya Deshpande, Jiajun Lu, Mao-Chuang Yeh, Min Jin Chong, and David Forsyth. Learning diverse image colorization. In *CVPR*, 2017. 8, 10

[11] S. R. Dubey, S. Chakraborty, S. K. Roy, S. Mukherjee, S. K. Singh, and B. B. Chaudhuri. diffGrad: An optimization method for convolutional neural networks. *IEEE*

Figure 19: Comparisons between our universal ReHistoGAN and the methods proposed by Shih et al., [31] and Laffont et al., [20] for color transfer.

Figure 20: Comparisons between our ReHistoGAN and the diverse colorization method proposed by Deshpande et al., [10]. For our ReHistoGAN, we show the resutls of our domain-specific and universal models.

Transactions on Neural Networks and Learning Systems, 31(11):4500–4511, 2020. 1, 4

[12] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 8

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 2

[15] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 3, 6, 7

[16] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020. 1, 5, 6, 7

[17] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *CVPR Workshops*, 2011. 1, 3

[18] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D object representations for fine-grained categorization. In *ICCV Workshops*, 2013. 1, 3

[19] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset V4. *International Journal of Computer Vision*, pages 1–26, 2020. 3

[20] Pierre-Yves Laffont, Zhile Ren, Xiaofeng Tao, Chao Qian, and James Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Transactions on graphics (TOG)*, 33(4):1–11, 2014. 8, 10

[21] Yuheng Li, Krishna Kumar Singh, Utkarsh Ojha, and Yong Jae Lee. MixNMatch: Multifactor disentanglement and encoding for conditional image generation. In *CVPR*, 2020. 5, 7

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence

Figure 21: Comparison with the auto image recoloring method proposed in [4]. Our recoloring results were produced by domain-specific networks, except for the last three rows, where our results were generated by the universal model.

Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 3

[23] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 3

[24] Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can semantic labeling methods generalize to any city? The inria aerial image labeling benchmark. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, 2017. 1

[25] Rang MH Nguyen, Seon Joo Kim, and Michael S Brown. Illuminant aware gamut-based color transfer. In *Computer Graphics Forum*, 2014. 8

[26] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008. 1, 3

[27] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *SIGGRAPH*. 2003. 3

[28] F. Pitie and A. Kokaram. The linear Monge-Kantorovitch linear colour mapping for example-based colour transfer. In *European Conference on Visual Media Production*, 2007. 4, 5, 8

[29] Erik Reinhard, Michael Adhikhmin, Bruce Gooch, and Peter Shirley. Color transfer between images. *IEEE Computer graphics and applications*, 21(5):34–41, 2001. 8

[30] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-Net: Multi-scale zero-shot style transfer by feature decoration. In *CVPR*, 2018. 8

[31] Yichang Shih, Sylvain Paris, Frédo Durand, and William T Freeman. Data-driven hallucination of different times of day from a single outdoor photo. *ACM Transactions on Graphics (TOG)*, 32(6):1–11, 2013. 10

[32] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD birds-200-2011 dataset. 2011. 1, 3

[33] Xuezhong Xiao and Lizhuang Ma. Color transfer in correlated color space. In *International conference on Virtual reality continuum and its applications*, 2006. 8

[34] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 1, 3

[35] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017. 3

[36] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019. 3