Figure 1. Examples from our newly collected AFHQ dataset.

A. The AFHQ dataset

We release a new dataset of animal faces, Animal Faces-HQ (AFHQ), consisting of 15,000 high-quality images at 512×512 resolution. Figure 1 shows example images of the AFHQ dataset. The dataset includes three domains of cat, dog, and wildlife, each providing 5000 images. By having multiple (three) domains and diverse images of various breeds ($\geq$ eight) per each domain, AFHQ sets a more challenging image-to-image translation problem. For each domain, we select 500 images as a test set and provide all remaining images as a training set. We collected images with permissive licenses from the Flickr[1] and Pixabay[2] websites. All images are vertically and horizontally aligned to have the eyes at the center. The low-quality images were discarded by human effort. We have made dataset available at https://github.com/clovaai/stargan-v2.

B. Training details

For fast training, the batch size is set to eight and the model is trained for 100K iterations. The training time is about three days on a single Tesla V100 GPU with our implementation in PyTorch [19]. We set $\lambda_{sty} = 1$, $\lambda_{ds} = 1$, and $\lambda_{cyc} = 1$ for CelebA-HQ and $\lambda_{sty} = 1$, $\lambda_{ds} = 2$, and $\lambda_{cyc} = 1$ for AFHQ. To stabilize the training, the weight λ_{ds} is linearly decayed to zero over the 100K iterations. We adopt the non-saturating adversarial loss [2] with R_1 regularization [15] using $\gamma = 1$. We use the Adam [12] optimizer with $\beta_1 = 0$ and $\beta_2 = 0.99$. The learning rates for G, D, and E are set to 10^{-4}, while that of F is set to 10^{-6}. For evaluation, we employ exponential moving averages over parameters [10, 23] of all modules except D. We initialize the weights of all modules using He initialization [3] and set all biases to zero, except for the biases associated with the scaling vectors of AdaIN that are set to one.

[1]https://www.flickr.com
[2]https://www.pixabay.com

C. Evaluation protocol

This section provides details for the evaluation metrics and evaluation protocols used in all experiments.

Frechét inception distance (FID) [5] measures the discrepancy between two sets of images. We use the feature vectors from the last average pooling layer of the ImageNet-pretrained Inception-V3 [21]. For each test image from a source domain, we translate it into a target domain using 10 latent vectors, which are randomly sampled from the standard Gaussian distribution. We then calculate FID between the translated images and training images in the target domain. We calculate the FID values for every pair of image domains (*e.g.* female $\rightleftarrows$ male for CelebA-HQ) and report the average value. Note that, for reference-guided synthesis, each source image is transformed using 10 reference images randomly sampled from the test set of a target domain.

Learned perceptual image patch similarity (LPIPS) [24] measures the diversity of generated images using the L_1 distance between features extracted from the ImageNet-pretrained AlexNet [13]. For each test image from a source domain, we generate 10 outputs of a target domain using 10 randomly sampled latent vectors. Then, we compute the average of the pairwise distances among all outputs generated from the same input (*i.e.* 45 pairs). Finally, we report the average of the LPIPS values over all test images. For reference-guided synthesis, each source image is transformed using 10 reference images to produce 10 outputs.

D. Additional results

We provide additional reference-guided image synthesis results on both CelebA-HQ and AFHQ (Figure 2 and 3). In CelebA-HQ, StarGAN v2 synthesizes the source identity in diverse appearances reflecting the reference styles such as hairstyle, makeup, and beard. In AFHQ, the result images follow the breed and hair of the reference images preserving the pose of the source images. Interpolation results between styles can be found in the accompanying videos.

Figure 2. Reference-guided image synthesis results on CelebA-HQ. The source and reference images in the first row and the first column are real images, while the rest are images generated by our proposed model, StarGAN v2. Our model learns to transform a source image reflecting the style of a given reference image. High-level semantics such as hairstyle, makeup, beard and age are followed from the reference images, while the pose and identity of the source images are preserved. Note that the images in each column share a single identity with different styles, and those in each row share a style with different identities.

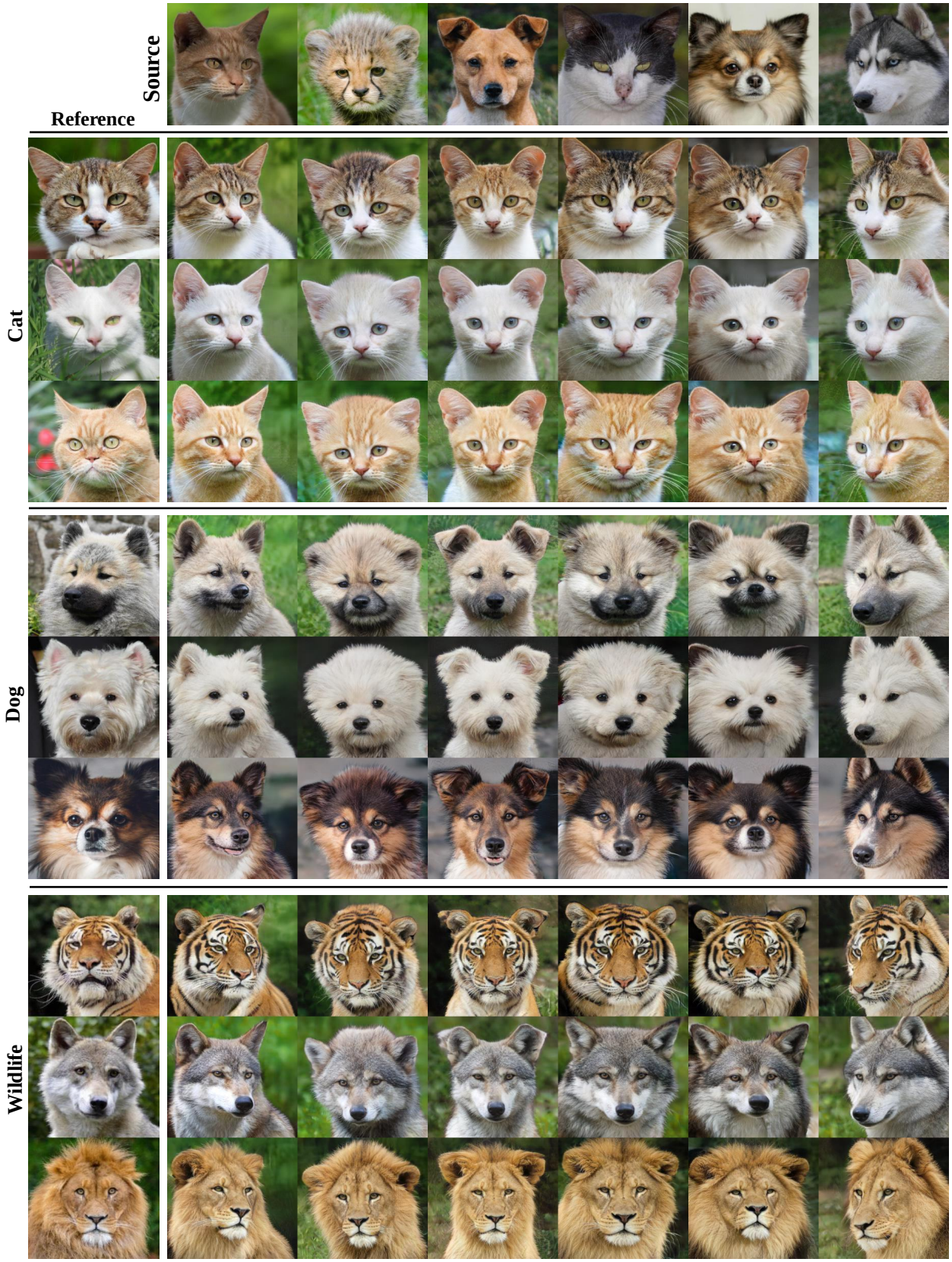

Figure 3. Reference-guided image synthesis results on AFHQ. All images except the sources and references are generated by our proposed model, StarGAN v2. High-level semantics such as hair are followed from the references, while the pose of the sources are preserved.

E. Network architecture

In this section, we provide architectural details of StarGAN v2, which consists of four modules described below.

Generator (Table 1)**.** Our generator consists of four down-sampling blocks, four intermediate blocks, and four up-sampling blocks, all of which inherit pre-activation residual units [4]. We use the instance normalization (IN) [22] and the adaptive instance normalization (AdaIN) [6, 11] for down-sampling and up-sampling blocks, respectively. A style code is injected into all AdaIN layers, providing scaling and shifting vectors through learned affine transformations. We use the average pooling for down-sampling and the nearest-neighbor interpolation for up-sampling. We do not use the hyperbolic tangent as an output activation and let the model to learn the output color range.

Mapping network (Table 2)**.** Our mapping network consists of an MLP with `K` output branches, where `K` indicates the number of domains. Four fully connected layers are shared among all domains, followed by four specific fully connected layers for each domain. We set the dimensions of the latent code, the hidden layer, and the style code to 16, 512, and 64, respectively. We sample the latent code from the standard Gaussian distribution. We do not apply the pixel normalization [11] to the latent code, which has been observed not to increase model performance in our tasks. We also tried feature normalizations [1, 8], but this degraded performance.

Style encoder (Table 3)**.** Our style encoder consists of a CNN with `K` output branches, where `K` is the number of domains. Six pre-activation residual blocks are shared among all domains, followed by one specific fully connected layer for each domain. We do not use the global average pooling [7] to extract fine style features of a given reference image. The output dimension "`D`" in Table 3 is set to 64, which indicates the dimension of the style code.

Discriminator (Table 3)**.** Our discriminator is a multi-task discriminator [15], which contains multiple linear output branches [3]. The discriminator contains six pre-activation residual blocks with leaky ReLU [14]. We use `K` fully-connected layers for real/fake classification of each domain, where `K` indicates the number of domains. The output dimension "`D`" is set to 1 for real/fake classification. We do not use any feature normalization techniques [8, 22] nor PatchGAN [9] as they have been observed not to improve output quality. We have observed that in our settings, the multi-task discriminator provides better results than other types of conditional discriminators [16, 17, 18, 20].

[3]The original implementation of the multi-task discriminator can be found at https://github.com/LMescheder/GAN_stability.

LAYER	RESAMPLE	NORM	OUTPUT SHAPE
Image $\mathbf{x}$	-	-	$256 \times 256 \times 3$
Conv1×1	-	-	$256 \times 256 \times 64$
ResBlk	AvgPool	IN	$128 \times 128 \times 128$
ResBlk	AvgPool	IN	$64 \times 64 \times 256$
ResBlk	AvgPool	IN	$32 \times 32 \times 512$
ResBlk	AvgPool	IN	$16 \times 16 \times 512$
ResBlk	-	IN	$16 \times 16 \times 512$
ResBlk	-	IN	$16 \times 16 \times 512$
ResBlk	-	AdaIN	$16 \times 16 \times 512$
ResBlk	-	AdaIN	$16 \times 16 \times 512$
ResBlk	Upsample	AdaIN	$32 \times 32 \times 512$
ResBlk	Upsample	AdaIN	$64 \times 64 \times 256$
ResBlk	Upsample	AdaIN	$128 \times 128 \times 128$
ResBlk	Upsample	AdaIN	$256 \times 256 \times 64$
Conv1×1	-	-	$256 \times 256 \times 3$

Table 1. Generator architecture.

TYPE	LAYER	ACTVATION	OUTPUT SHAPE
Shared	Latent $\mathbf{z}$	-	16
Shared	Linear	ReLU	512
Shared	Linear	ReLU	512
Shared	Linear	ReLU	512
Shared	Linear	ReLU	512
Unshared	Linear	ReLU	512
Unshared	Linear	ReLU	512
Unshared	Linear	ReLU	512
Unshared	Linear	-	64

Table 2. Mapping network architecture.

LAYER	RESAMPLE	NORM	OUTPUT SHAPE
Image $\mathbf{x}$	-	-	$256 \times 256 \times 3$
Conv1×1	-	-	$256 \times 256 \times 64$
ResBlk	AvgPool	-	$128 \times 128 \times 128$
ResBlk	AvgPool	-	$64 \times 64 \times 256$
ResBlk	AvgPool	-	$32 \times 32 \times 512$
ResBlk	AvgPool	-	$16 \times 16 \times 512$
ResBlk	AvgPool	-	$8 \times 8 \times 512$
ResBlk	AvgPool	-	$4 \times 4 \times 512$
LReLU	-	-	$4 \times 4 \times 512$
Conv4×4	-	-	$1 \times 1 \times 512$
LReLU	-	-	$1 \times 1 \times 512$
Reshape	-	-	512
Linear $\star$ K	-	-	D $\star$ K

Table 3. Style encoder and discriminator architectures. `D` and `K` represent the output dimension and number of domains, respectively.

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. In *arXiv preprint*, 2016. 4

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *NeurIPS*, 2014. 1

[3] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 1

[4] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 4

[5] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 1

[6] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 4

[7] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 4

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 4

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial nets. In *CVPR*, 2017. 4

[10] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 1

[11] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 4

[12] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 1

[14] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 4

[15] L. Mescheder, S. Nowozin, and A. Geiger. Which training methods for gans do actually converge? In *ICML*, 2018. 1, 4

[16] M. Mirza and S. Osindero. Conditional generative adversarial nets. In *arXiv preprint*, 2014. 4

[17] T. Miyato and M. Koyama. cGANs with projection discriminator. In *ICLR*, 2018. 4

[18] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, 2017. 4

[19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NeurIPSW*, 2017. 1

[20] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *ICML*, 2016. 4

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 1

[22] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. In *arXiv preprint*, 2016. 4

[23] Y. Yazıcı, C.-S. Foo, S. Winkler, K.-H. Yap, G. Piliouras, and V. Chandrasekhar. The unusual effectiveness of averaging in gan training. In *ICLR*, 2019. 1

[24] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 1