arXiv:1912.01865v2 [cs.CV] 26 Apr 2020

StarGAN v2: Diverse Image Synthesis for Multiple Domains

Yunjey Choi[1*] Youngjung Uh[1*] Jaejun Yoo[2*] Jung-Woo Ha[1]

[1]Clova AI Research, NAVER Corp. [2]EPFL

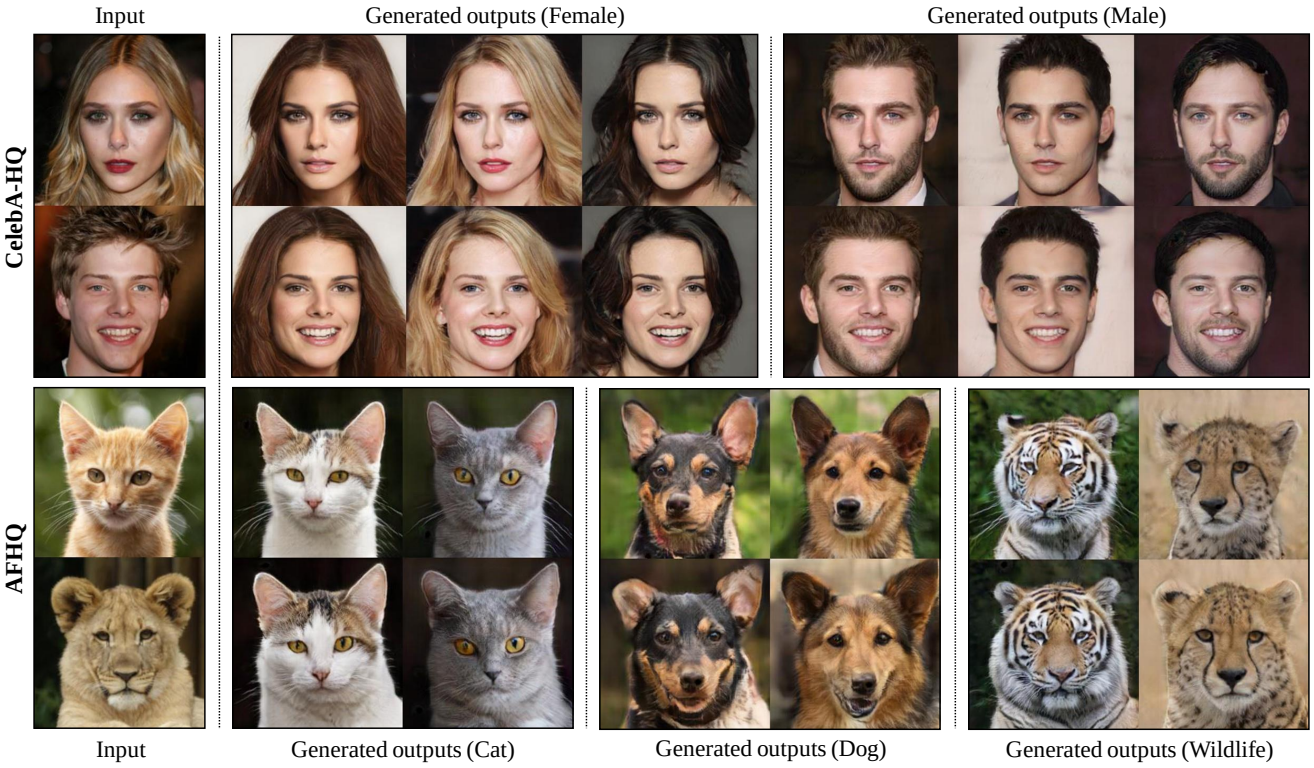


Figure 1. Diverse image synthesis results on the CelebA-HQ dataset and the newly collected animal faces (AFHQ) dataset. The first column shows input images while the remaining columns are images synthesized by StarGAN v2.

Abstract

A good image-to-image translation model should learn a mapping between different visual domains while satisfying the following properties: 1) diversity of generated images and 2) scalability over multiple domains. Existing methods address either of the issues, having limited diversity or multiple models for all domains. We propose StarGAN v2, a single framework that tackles both and shows significantly improved results over the baselines. Experiments on CelebA-HQ and a new animal faces dataset (AFHQ) validate our superiority in terms of visual quality, diversity, and scalability. To better assess image-to-image translation models, we release AFHQ, high-quality animal faces with large inter- and intra-domain differences. The code, pretrained models, and dataset are available at clovaai/stargan-v2.

1. Introduction

Image-to-image translation aims to learn a mapping between different visual domains [20]. Here, `domain` implies a set of images that can be grouped as a visually distinctive category, and each image has a unique appearance, which we call `style`. For example, we can set image domains based on the gender of a person, in which case the style includes makeup, beard, and hairstyle (top half of Figure 1). An ideal image-to-image translation method should be able to synthesize images considering the diverse styles in each domain. However, designing and learning such models become complicated as there can be arbitrarily large number of styles and domains in the dataset.

e.g.
Domain ~ Gender
Style ~ Long Hair

* indicates equal contribution

Work done so far...

To address the style diversity, much work on image-to-image translation has been developed [1, 16, 34, 28, 38, 54]. These methods inject a low-dimensional latent code to the generator, which can be randomly sampled from the standard Gaussian distribution. Their domain-specific decoders interpret the latent codes as recipes for various styles when generating images. However, because these methods have only considered a mapping between two domains, they are not scalable to the increasing number of domains. For example, having `K` domains, these methods require to train `K(K-1)` generators to handle translations between each and every domain, limiting their practical usage.

To address the scalability, several studies have proposed a unified framework [2, 7, 17, 30]. StarGAN [7] is one of the earliest models, which learns the mappings between all available domains using a single generator. The generator takes a domain label as an additional input, and learns to transform an image into the corresponding domain. However, StarGAN still learns a deterministic mapping per each domain, which does not capture the multi-modal nature of the data distribution. This limitation comes from the fact that each domain is indicated by a predetermined label. Note that the generator receives a fixed label (*e.g.* one-hot vector) as input, and thus it inevitably produces the same output per each domain, given a source image.

To get the best of both worlds, we propose StarGAN v2, a scalable approach that can generate **diverse** images across **multiple domains**. In particular, we start from StarGAN and replace its domain label with our proposed domain-specific style code that can represent diverse `styles` of a specific `domain`. To this end, we introduce two modules, a mapping network and a style encoder. The mapping network learns to transform random Gaussian noise into a style code, while the encoder learns to extract the style code from a given reference image. Considering multiple domains, both modules have multiple output branches, each of which provides style codes for a specific domain. Finally, utilizing these style codes, our generator learns to successfully synthesize diverse images over multiple domains (Figure 1).

We first investigate the effect of individual components of StarGAN v2 and show that our model indeed benefits from using the style code (Section 3.1). We empirically demonstrate that our proposed method is scalable to multiple domains and gives significantly better results in terms of visual quality and diversity compared to the leading methods (Section 3.2). Last but not least, we present a new dataset of animal faces (AFHQ) with high quality and wide variations (Appendix A) to better evaluate the performance of image-to-image translation models on large inter- and intra-domain differences. We release this dataset publicly available for research community.

2. StarGAN v2

In this section, we describe our proposed framework and its training objective functions.

2.1. Proposed framework

Let $\mathcal{X}$ and $\mathcal{Y}$ be the sets of images and possible domains, respectively. Given an image $\mathbf{x} \in \mathcal{X}$ and an arbitrary domain $y \in \mathcal{Y}$, our goal is to train a **single** generator G that can generate **diverse** images of each domain y that corresponds to the image $\mathbf{x}$. We generate **domain-specific** style vectors in the learned style space of each domain and train G to reflect the style vectors. Figure 2 illustrates an overview of our framework, which consists of four modules described below.

Generator (Figure 2a)**.** Our generator G translates an input image $\mathbf{x}$ into an output image $G(\mathbf{x}, \mathbf{s})$ reflecting a domain-specific style code $\mathbf{s}$, which is provided either by the mapping network F or by the style encoder E. We use adaptive instance normalization (AdaIN) [15, 22] to inject $\mathbf{s}$ into G. We observe that $\mathbf{s}$ is designed to represent a style of a specific domain y, which removes the necessity of providing y to G and allows G to synthesize images of all domains.

Mapping network (Figure 2b)**.** Given a latent code $\mathbf{z}$ and a domain y, our mapping network F generates a style code $\mathbf{s} = F_y(\mathbf{z})$, where $F_y(\cdot)$ denotes an output of F corresponding to the domain y. F consists of an MLP with multiple output branches to provide style codes for all available domains. F can produce diverse style codes by sampling the latent vector $\mathbf{z} \in \mathcal{Z}$ and the domain $y \in \mathcal{Y}$ randomly. Our multi-task architecture allows F to efficiently and effectively learn style representations of all domains.

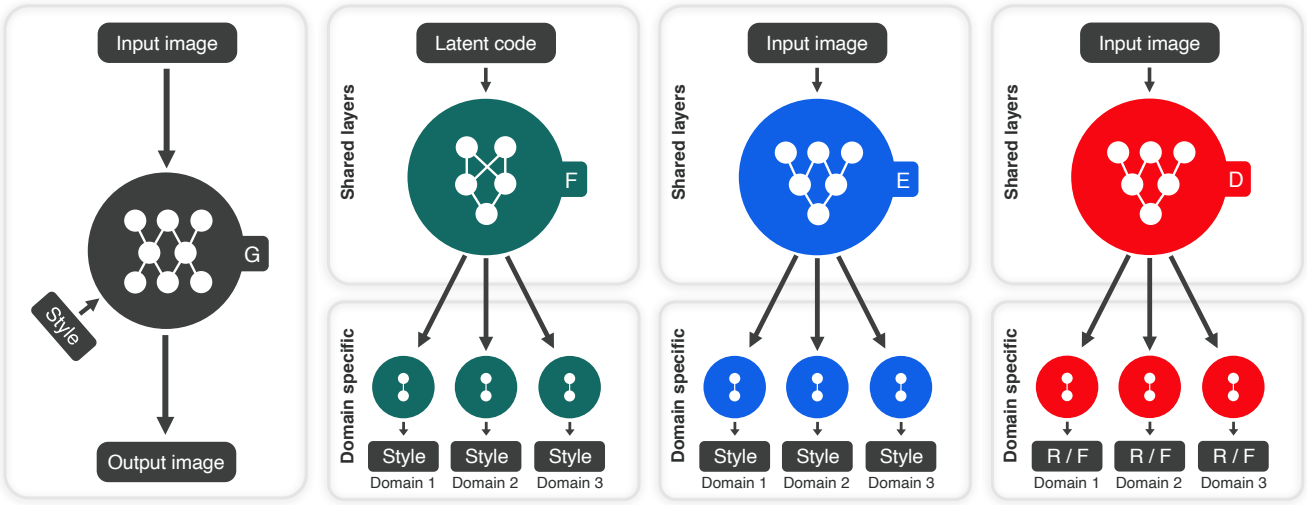
Style encoder (Figure 2c)**.** Given an image $\mathbf{x}$ and its corresponding domain y, our encoder E extracts the style code $\mathbf{s} = E_y(\mathbf{x})$ of $\mathbf{x}$. Here, $E_y(\cdot)$ denotes the output of E corresponding to the domain y. Similar to F, our style encoder E benefits from the multi-task learning setup. E can produce diverse style codes using different reference images. This allows G to synthesize an output image reflecting the style $\mathbf{s}$ of a reference image $\mathbf{x}$.

Discriminator (Figure 2d)**.** Our discriminator D is a multi-task discriminator [30, 35], which consists of multiple output branches. Each branch D_y learns a binary classification determining whether an image $\mathbf{x}$ is a real image of its domain y or a fake image $G(\mathbf{x}, \mathbf{s})$ produced by G.

2.2. Training objectives

Given an image $\mathbf{x} \in \mathcal{X}$ and its original domain $y \in \mathcal{Y}$, we train our framework using the following objectives.

Adversarial objective. During training, we sample a latent code $\mathbf{z} \in \mathcal{Z}$ and a target domain $\tilde{y} \in \mathcal{Y}$ randomly, and



(a) Generator (b) Mapping network (c) Style encoder (d) Discriminator

Figure 2. Overview of StarGAN v2, consisting of four modules. **(a)** The generator translates an input image into an output image reflecting the domain-specific style code. **(b)** The mapping network transforms a latent code into style codes for multiple domains, one of which is randomly selected during training. **(c)** The style encoder extracts the style code of an image, allowing the generator to perform reference-guided image synthesis. **(d)** The discriminator distinguishes between real and fake images from multiple domains. Note that all modules except the generator contain multiple output branches, one of which is selected when training the corresponding domain.

generate a target style code $\widetilde{\mathbf{s}} = F_{\widetilde{y}}(\mathbf{z})$. The generator G takes an image $\mathbf{x}$ and $\widetilde{\mathbf{s}}$ as inputs and learns to generate an output image $G(\mathbf{x}, \widetilde{\mathbf{s}})$ via an adversarial loss

G (x, s~) is intended to be in y~.

x is actually in y. So, Dy(x) should be true

$$\mathcal{L}_{adv} = \mathbb{E}_{\mathbf{x},y}\left[\log D_y(\mathbf{x})\right] + \mathbb{E}_{\mathbf{x},\widetilde{y},\mathbf{z}}[\log\left(1 - D_{\widetilde{y}}(G(\mathbf{x}, \widetilde{\mathbf{s}}))\right)], \quad (1)$$

intended to be True

where $D_y(\cdot)$ denotes the output of D corresponding to the domain y. The mapping network F learns to provide the style code $\widetilde{\mathbf{s}}$ that is likely in the target domain $\widetilde{y}$, and G learns to utilize $\widetilde{\mathbf{s}}$ and generate an image $G(\mathbf{x}, \widetilde{\mathbf{s}})$ that is indistinguishable from real images of the domain $\widetilde{y}$.

Style reconstruction. In order to enforce the generator G to utilize the style code $\widetilde{\mathbf{s}}$ when generating the image $G(\mathbf{x}, \widetilde{\mathbf{s}})$, we employ a style reconstruction loss

$$\mathcal{L}_{sty} = \mathbb{E}_{\mathbf{x},\widetilde{y},\mathbf{z}}\left[||\widetilde{\mathbf{s}} - E_{\widetilde{y}}(G(\mathbf{x}, \widetilde{\mathbf{s}}))||_1\right]. \quad (2)$$

This objective is similar to the previous approaches [16, 54], which employ multiple encoders to learn a mapping from an image to its latent code. The notable difference is that we train a **single** encoder E to encourage diverse outputs for multiple domains. At test time, our learned encoder E allows G to transform an input image, reflecting the style of a reference image.

E represents the Encoder, which finds the style of a given image;

So while inverting to get the style of the generated image, we should get back the style s~ which was intended...

Style diversification. To further enable the generator G to produce diverse images, we explicitly regularize G with the diversity sensitive loss [34, 48]

$$\mathcal{L}_{ds} = \mathbb{E}_{\mathbf{x},\widetilde{y},\mathbf{z}_1,\mathbf{z}_2}\left[\|G(\mathbf{x}, \widetilde{\mathbf{s}}_1) - G(\mathbf{x}, \widetilde{\mathbf{s}}_2)\|_1\right], \quad (3)$$

where the target style codes $\widetilde{\mathbf{s}}_1$ and $\widetilde{\mathbf{s}}_2$ are produced by F conditioned on two random latent codes $\mathbf{z}_1$ and $\mathbf{z}_2$ (*i.e.* $\widetilde{\mathbf{s}}_i = F_{\widetilde{y}}(\mathbf{z}_i)$ for $i \in \{1, 2\}$). Maximizing the regularization term forces G to explore the image space and discover meaningful style features to generate diverse images. Note that in the original form, the small difference of $\|\mathbf{z}_1 - \mathbf{z}_2\|_1$ in the denominator increases the loss significantly, which makes the training unstable due to large gradients. Thus, we remove the denominator part and devise a new equation for stable training but with the same intuition.

TO BE MAXIMISED... (regularisation term)

Preserving source characteristics. To guarantee that the generated image $G(\mathbf{x}, \widetilde{\mathbf{s}})$ properly preserves the domain-invariant characteristics (*e.g.* pose) of its input image $\mathbf{x}$, we employ the cycle consistency loss [7, 24, 53]

$$\mathcal{L}_{cyc} = \mathbb{E}_{\mathbf{x},y,\widetilde{y},\mathbf{z}}\left[||\mathbf{x} - G(G(\mathbf{x}, \widetilde{\mathbf{s}}), \hat{\mathbf{s}})||_1\right], \quad (4)$$

^ s is in the original domain, while s ~ is in the domain y~ in which the new image was formed

where $\hat{\mathbf{s}} = E_y(\mathbf{x})$ is the estimated style code of the input image $\mathbf{x}$, and y is the original domain of $\mathbf{x}$. By encouraging the generator G to reconstruct the input image $\mathbf{x}$ with the estimated style code $\hat{\mathbf{s}}$, G learns to preserve the original characteristics of $\mathbf{x}$ while changing its style faithfully.

Full objective. Our full objective functions can be summarized as follows:

$$\min_{G,F,E} \max_{D} \mathcal{L}_{adv} + \lambda_{sty}\mathcal{L}_{sty} - \lambda_{ds}\mathcal{L}_{ds} + \lambda_{cyc}\mathcal{L}_{cyc}, \quad (5)$$

where λ_{sty}, λ_{ds}, and λ_{cyc} are hyperparameters for each term. We also further train our model in the same manner as the above objective, using reference images instead of latent vectors when generating style codes. We provide the training details in Appendix B.

	Method	FID	LPIPS
A	Baseline StarGAN [7]	98.4	-
B	+ Multi-task discriminator	91.4	-
C	+ Tuning (*e.g.* R_1 regularization)	80.5	-
D	+ Latent code injection	32.3	0.312
E	+ Replace (D) with style code	17.1	0.405
F	+ Diversity regularization	**13.7**	**0.452**

Table 1. Performance of various configurations on CelebA-HQ. Frechet inception distance (FID) indicates the distance between two distributions of real and generated images (lower is better), while learned perceptual image patch similarity (LPIPS) measures the diversity of generated images (higher is better).

3. Experiments

In this section, we describe evaluation setups and conduct a set of experiments. We analyze the individual components of StarGAN v2 (Section 3.1) and compare our model with three leading baselines on diverse image synthesis (Section 3.2). All experiments are conducted using unseen images during the training phase.

Baselines. We use MUNIT [16], DRIT [28], and MSGAN [34] as our baselines, all of which learn multi-modal mappings between two domains. For multi-domain comparisons, we train these models multiple times for every pair of image domains. We also compare our method with StarGAN [7], which learns mappings among multiple domains using a single generator. All the baselines are trained using the implementations provided by the authors.

Datasets. We evaluate StarGAN v2 on CelebA-HQ [21] and our new AFHQ dataset (Appendix A). We separate CelebA-HQ into two domains of male and female, and AFHQ into three domains of cat, dog, and wildlife. Other than the domain labels, we do not use any additional information (*e.g.* facial attributes of CelebA-HQ or breeds of AFHQ) and let the models learn such information as styles without supervision. For a fair comparison, all images are resized to 256×256 resolution for training, which is the highest resolution used in the baselines.

Evaluation metrics. We evaluate both the visual quality and the diversity of generated images using Frechét inception distance (FID) [14] and learned perceptual image patch similarity (LPIPS) [52]. We compute FID and LPIPS for every pair of image domains within a dataset and report their average values. The details on evaluation metrics and protocols are further described in Appendix C.

3.1. Analysis of individual components

We evaluate individual components that are added to our baseline StarGAN using CelebA-HQ. Table 1 gives FID and LPIPS for several configurations, where each component is cumulatively added on top of StarGAN. An input image and the corresponding generated images of each configuration are shown in Figure 3. The baseline configuration (A) corresponds to the basic setup of StarGAN, which employs WGAN-GP [11], ACGAN discriminator [39], and depth-wise concatenation [36] for providing the target domain information to the generator. As shown in Figure 3a, the original StarGAN produces only a local change by applying makeup on the input image.

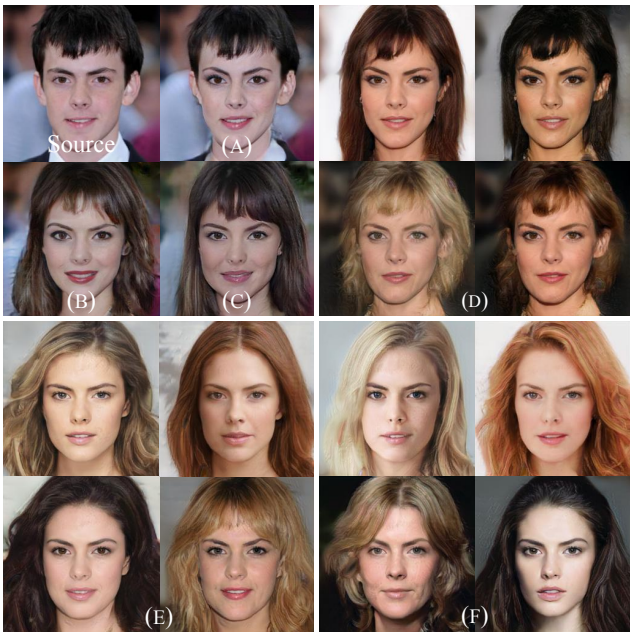


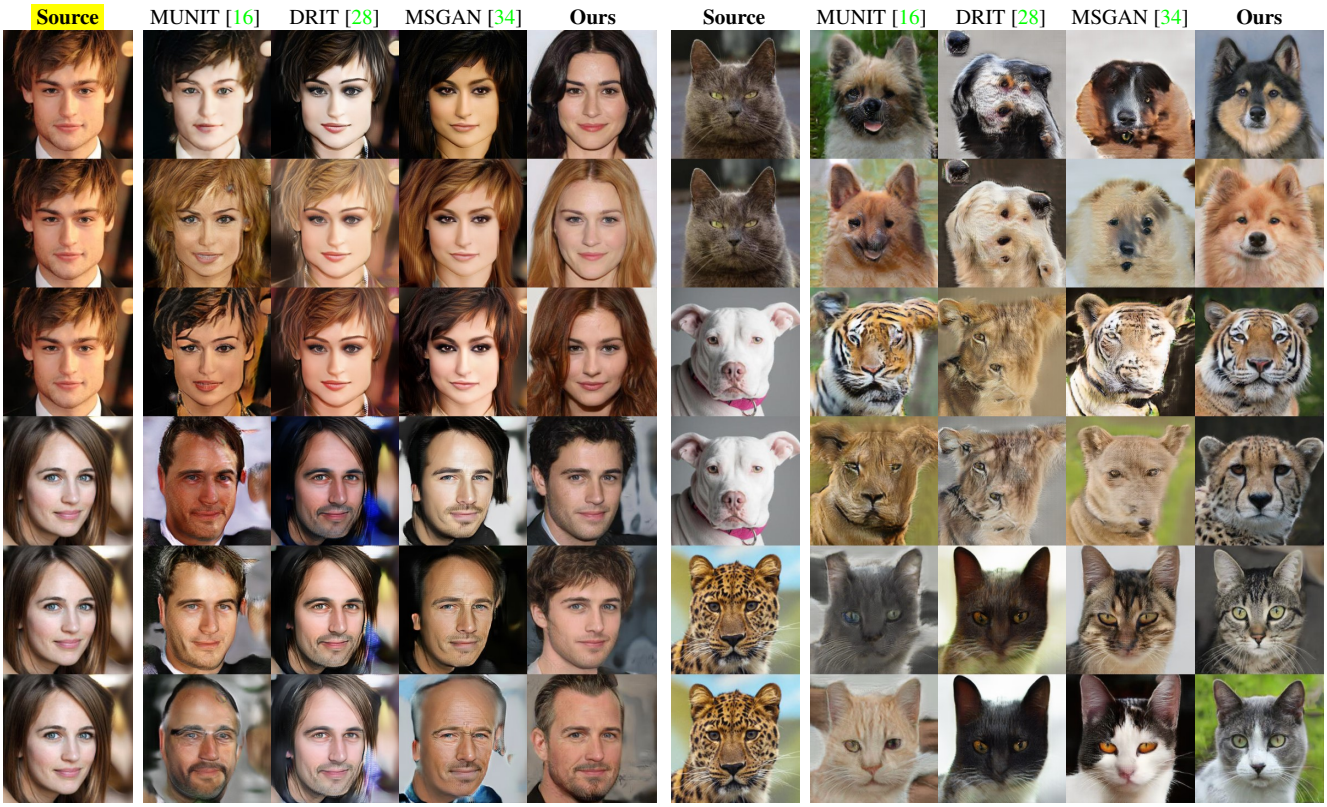
Figure 3. Visual comparison of generated images using each configuration in Table 1. Note that given a source image, the configurations (A) - (C) provide a single output, while (D) - (F) generate multiple output images.

We first improve our baseline by replacing the ACGAN discriminator with a multi-task discriminator [35, 30], allowing the generator to transform the global structure of an input image as shown in configuration (B). Exploiting the recent advances in GANs, we further enhance the training stability and construct a new baseline (C) by applying R_1 regularization [35] and switching the depth-wise concatenation to adaptive instance normalization (AdaIN) [9, 15]. Note that we do not report LPIPS of these variations in Table 1, since they are yet to be designed to produce multiple outputs for a given input image and a target domain.

To induce diversity, one can think of directly giving a latent code $\mathbf{z}$ into the generator G and impose the latent reconstruction loss $||\mathbf{z} - E(G(\mathbf{x}, \mathbf{z}, y))||_1$ [16, 54]. However, in a *multi-domain* scenario, we observe that this baseline (D) does not encourage the network to learn meaningful styles and fails to provide as much diversity as we expect. We conjecture that this is because latent codes have no capability in separating domains, and thus the latent reconstruction loss models *domain-shared* styles (*e.g.* color) rather than *domain-specific* ones (*e.g.* hairstyle). Note that the FID gap between baseline (C) and (D) is simply due to the difference in the number of output samples.

Figure 4. 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.

Transforms source image with the style of reference image



(a) Latent-guided synthesis on CelebA-HQ (b) Latent-guided synthesis on AFHQ

Figure 5. Qualitative comparison of latent-guided image synthesis results on the CelebA-HQ and AFHQ datasets. Each method translates the source images (left-most column) to target domains using randomly sampled latent codes. **(a)** The top three rows correspond to the results of converting male to female and vice versa in the bottom three rows. **(b)** Every two rows from the top show the synthesized images in the following order: cat-to-dog, dog-to-wildlife, and wildlife-to-cat.

Instead of giving a latent code into G directly, to learn meaningful styles, we transform a latent code $\mathbf{z}$ into a *domain-specific* style code $\mathbf{s}$ through our proposed mapping network (Figure 2b) and inject the style code into the generator (E). Here, we also introduce the style reconstruction loss (Eq. (2)). Note that each output branch of our mapping network is responsible to a particular domain, thus style codes have no ambiguity in separating domains. Unlike the latent reconstruction loss, the style reconstruction loss allows the generator to produce diverse images reflecting *domain-specific* styles. Finally, we further improve the network to produce diverse outputs by adopting the diversity regularization (Eq. (3)), and this configuration (F) corresponds to our proposed method, StarGAN v2. Figure 4 shows that StarGAN v2 can synthesize images that reflect diverse styles of references including hairstyle, makeup, and beard, without hurting the source characteristics.

3.2. Comparison on diverse image synthesis

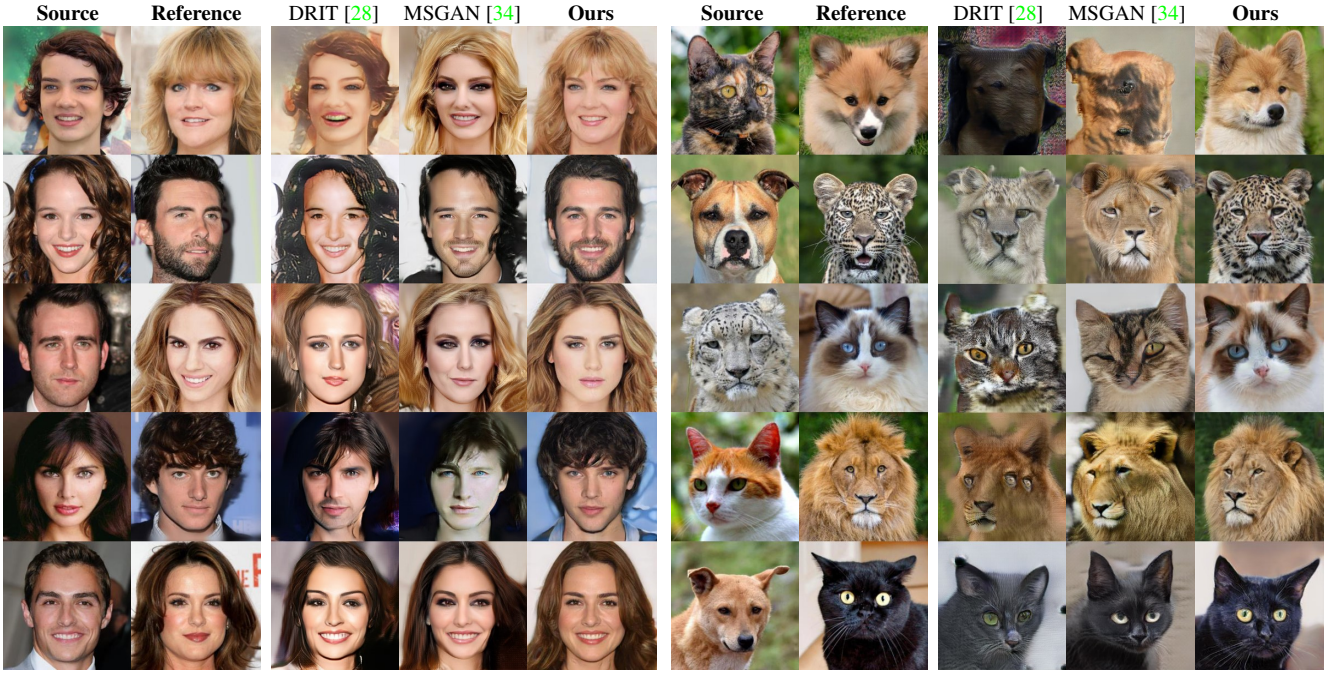
In this section, we evaluate StarGAN v2 on diverse image synthesis from two perspectives: latent-guided synthesis and reference-guided synthesis.

Latent-guided synthesis. Figure 5 provides a qualitative comparison of the competing methods. Each method produces multiple outputs using random noise. For CelebA-HQ, we observe that our method synthesizes images with a higher visual quality compared to the baseline models. In addition, our method is the only model that can successfully change the entire hair styles of the source images, which requires non-trivial effort (*e.g.* generating ears). For AFHQ, which has relatively large variations, the performance of the baselines is considerably degraded, while our method still produces images with high quality and diverse styles.

Method	CelebA-HQ		AFHQ	
	FID	LPIPS	FID	LPIPS
MUNIT [16]	31.4	0.363	41.5	0.511
DRIT [28]	52.1	0.178	95.6	0.326
MSGAN [34]	33.1	0.389	61.4	**0.517**
StarGAN v2	**13.7**	**0.452**	**16.2**	0.450
Real images	14.8	-	12.9	-

Table 2. Quantitative comparison on latent-guided synthesis. The FIDs of real images are computed between the training and test sets. Note that they may not be optimal values since the number of test images is insufficient, but we report them for reference.

As shown in Table 2, our method outperforms all the baselines by a large margin in terms of visual quality. For both CelebA-HQ and AFHQ, our method achieves FIDs of 13.7 and 16.2, respectively, which are more than two times



(a) Reference-guided synthesis on CelebA-HQ (b) Reference-guided synthesis on AFHQ

Figure 6. Qualitative comparison of reference-guided image synthesis results on the CelebA-HQ and AFHQ datasets. Each method translates the source images into target domains, reflecting the styles of the reference images.

improvement over the previous leading method. Our LPIPS is also the highest in CelebA-HQ, which implies our model produces the most diverse results given a single input. We conjecture that the high LPIPS values of the baseline models in AFHQ are due to their spurious artifacts.

Reference-guided synthesis. To obtain the style code from a reference image, we sample test images from a target domain and feed them to the encoder network of each method. For CelebA-HQ (Figure 6a), our method successfully renders distinctive styles (*e.g.* bangs, beard, makeup, and hairstyle), while the others mostly match the color distribution of reference images. For the more challenging AFHQ (Figure 6b), the baseline models suffer from a large domain shift. They hardly reflect the style of each reference image and only match the domain. In contrast, our model renders distinctive styles (*e.g.* breeds) of each reference image as well as its fur pattern and eye color. Note that StarGAN v2 produces high quality images across all domains and these results are from a single generator. Since the other baselines are trained individually for each pair of domains, the output quality fluctuates across domains. For example, in AFHQ (Figure 6b), the baseline models work reasonably well in dog-to-wildlife (2nd row) while they fail in cat-to-dog (1st row).

Table 3 shows FID and LPIPS of each method for reference guided synthesis. For both datasets, our method achieves FID of 23.8, and 19.8, which are about 1.5× and 3.5× better than the previous leading method, respectively.

Method	CelebA-HQ		AFHQ	
	FID	LPIPS	FID	LPIPS
MUNIT [16]	107.1	0.176	223.9	0.199
DRIT [28]	53.3	0.311	114.8	0.156
MSGAN [34]	39.6	0.312	69.8	0.375
StarGAN v2	**23.8**	**0.388**	**19.8**	**0.432**
Real images	14.8	-	12.9	-

Table 3. Quantitative comparison on reference-guided synthesis. We sample ten reference images to synthesize diverse images.

The LPIPS of StarGAN v2 is also the highest among the competitors, which implies that our model produces the most diverse results considering the styles of reference images. Here, MUNIT and DRIT suffer from mode-collapse in AFHQ, which results in lower LPIPS and higher FID than other methods.

Human evaluation. We use the Amazon Mechanical Turk (AMT) to compare the user preferences of our method with baseline approaches. Given a pair of source and reference images, the AMT workers are instructed to select one among four image candidates from the methods, whose order is randomly shuffled. We ask separately which model offers the best image quality and which model best stylizes the input image considering the reference image. For each comparison, we randomly generate 100 questions, and each question is answered by 10 workers. We also ask each worker a few simple questions to detect unworthy workers. The number of total valid workers is 76. As shown in Table 4, our method obtains the majority of votes in all in-

Method	CelebA-HQ		AFHQ	
	Quality	Style	Quality	Style
MUNIT [16]	6.2	7.4	1.6	0.2
DRIT [28]	11.4	7.6	4.1	2.8
MSGAN [34]	13.5	10.1	6.2	4.9
StarGAN v2	**68.9**	**74.8**	**88.1**	**92.1**

Table 4. Votes from AMT workers for the most preferred method regarding visual quality and style reflection (%). StarGAN v2 outperforms the baselines with remarkable margins in all aspects.

stances, especially in the challenging AFHQ dataset and the question about style reflection. These results show that StarGAN v2 better extracts and renders the styles onto the input image than the other baselines.

4. Discussion

We discuss several reasons why StarGAN v2 can successfully synthesize images of diverse styles over multiple domains. First, our style code is separately generated per domain by the multi-head mapping network and style encoder. By doing so, our generator can only focus on using the style code, whose domain-specific information is already taken care of by the mapping network (Section 3.1). Second, following the insight of StyleGAN [22], our style space is produced by learned transformations. This provides more flexibility to our model than the baselines [16, 28, 34], which assume that the style space is a fixed Gaussian distribution (Section 3.2). Last but not least, our modules benefit from fully exploiting training data from multiple domains. By design, the shared part of each module should learn domain-invariant features which induces the regularization effect, encouraging better generalization to unseen samples. To show that our model generalizes over the unseen images, we test a few samples from FFHQ [22] with our model trained on CelebA-HQ (Figure 7). Here, StarGAN v2 successfully captures styles of references and renders these styles correctly to the source images.

5. Related work

Generative adversarial networks (GANs) [10] have shown impressive results in many computer vision tasks such as image synthesis [4, 31, 8], colorization [18, 50] and super-resolution [27, 47]. Along with improving the visual quality of generated images, their diversity also has been considered as an important objective which has been tackled by either devoted loss functions [34, 35] or architectural design [4, 22]. StyleGAN [22] introduces a non-linear mapping function that embeds an input latent code into an intermediate style space to better represent the factors of variation. However, this method requires non-trivial effort when transforming a real image, since its generator is not designed to take an image as input.

Early image-to-image translation methods [20, 53, 29] are well known to learn a deterministic mapping even with stochastic noise inputs. Several methods reinforce the connection between stochastic noise and the generated image for diversity, by marginal matching [1], latent regression [54, 16], and diversity regularization [48, 34]. Other approaches produce various outputs with the guidance of reference images [5, 6, 32, 40]. However, all theses methods consider only two domains, and their extension to multiple domains is non-trivial. Recently, FUNIT [30] tackles multi-domain image translation using a few reference images from a target domain, but it requires fine-grained class labels and can not generate images with random noise. Our method provides both latent-guided and reference-guided synthesis and can be trained with coarsely labeled dataset. In parallel work, Yu et al. [51] tackle the same issue but they define the style as domain-shared characteristics rather than domain-specific ones, which limits the output diversity.

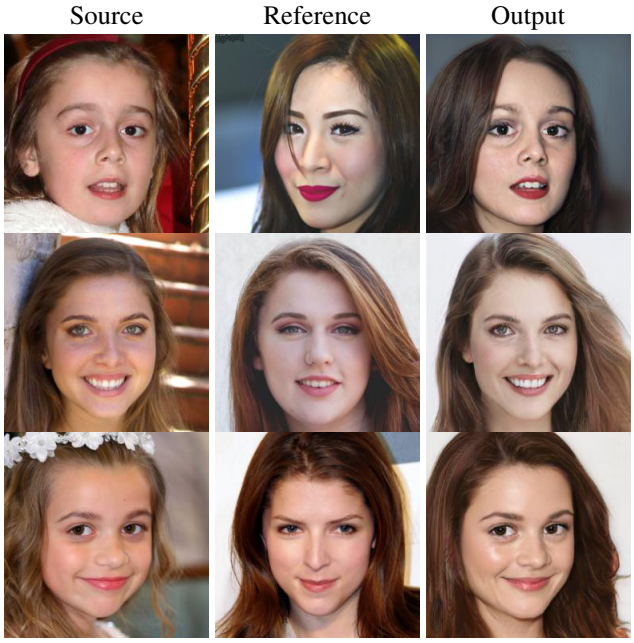


Figure 7. Reference-guided synthesis results on FFHQ with the model trained on CelebA-HQ. Despite the distribution gap between the two datasets, StarGAN v2 successfully extracts the style codes of the references and synthesizes faithful images.

6. Conclusion

We proposed StarGAN v2, which addresses two major challenges in image-to-image translation; translating an image of one domain to diverse images of a target domain, and supporting multiple target domains. The experimental results showed that our model can generate images with rich styles across multiple domains, remarkably outperforming the previous leading methods [16, 28, 34]. We also released a new dataset of animal faces (AFHQ) for evaluating methods in a large inter- and intra domain variation setting.

Acknowledgements. We thank the full-time and visiting Clova AI members for an early review: Seongjoon Oh, Junsuk Choe, Muhammad Ferjad Naeem, and Kyungjune Baek. All experiments were conducted based on NAVER Smart Machine Learning (NSML) [23, 43].

In StarGANv2 styles are domain specific characteristics

Generated images of different style for the same domain...

Figure 8. Examples from our newly collected AFHQ dataset. (Just the dataset; no training/ results shown here)

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 8 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 [41]. 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 [10] with R_1 regularization [35] using $\gamma = 1$. We use the Adam [25] 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 [21, 49] of all modules except D. We initialize the weights of all modules using He initialization [12] and set all biases to zero, except for the biases associated with the scaling vectors of AdaIN that are set to one.

Hyper-parameters

[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) [14] 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 [44]. 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.

Which signify the style vectors...

Learned perceptual image patch similarity (LPIPS) [52] measures the diversity of generated images using the L_1 distance between features extracted from the ImageNet-pretrained AlexNet [26]. 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.

We try to maximise LPIPS right??

D. Additional results

We provide additional reference-guided image synthesis results on both CelebA-HQ and AFHQ (Figure 9 and 10). In CelebA-HQ, StarGAN v2 synthesizes the source identity in diverse appearances reflecting the reference styles such as hairstyle, and makeup. In AFHQ, the results follow the breed and hair of the reference images preserving the pose of the source images. Interpolation results between styles can be found at https://youtu.be/0EVh5Ki4dIY.

Figure 9. 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.

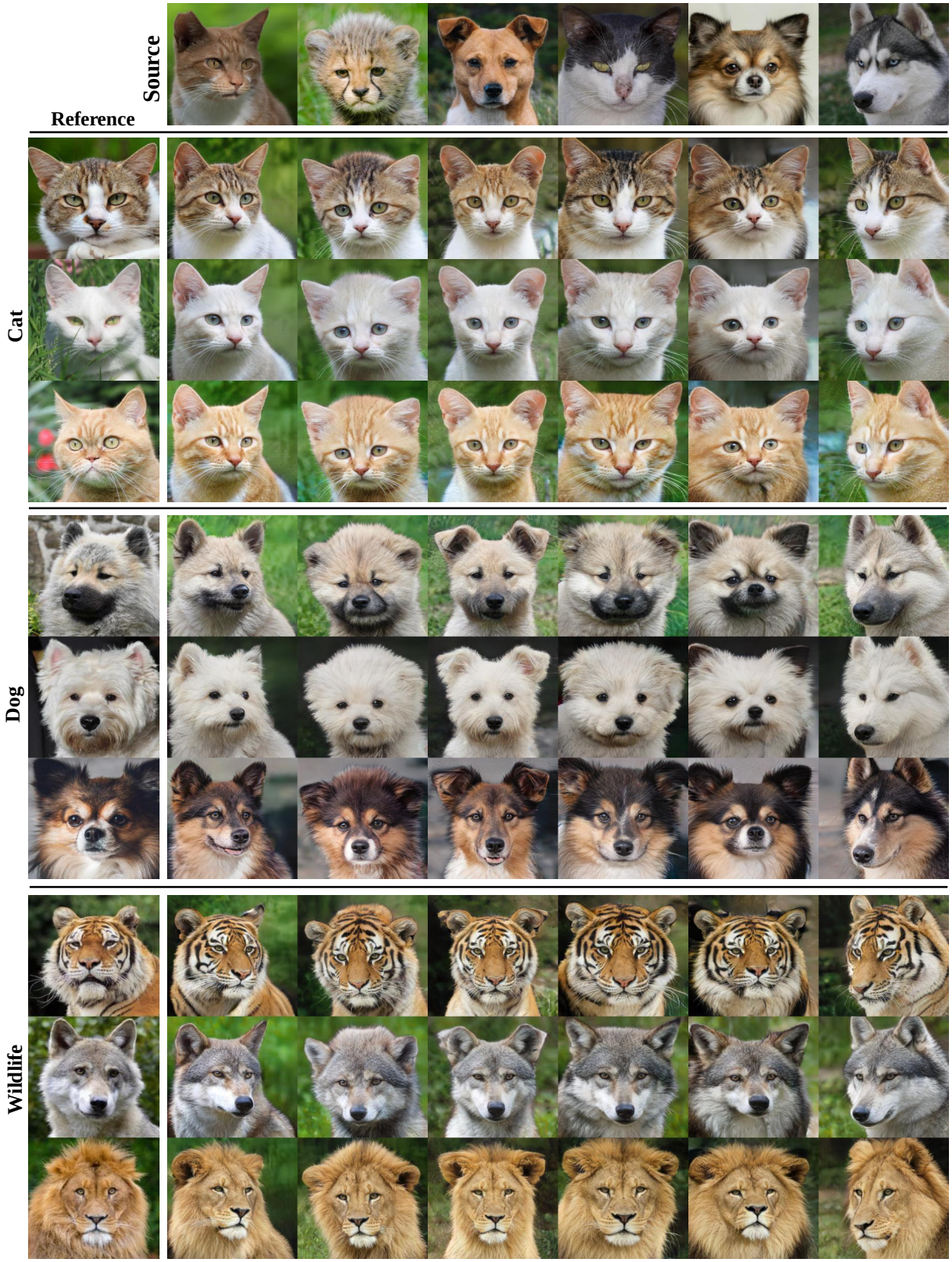


Figure 10. Reference-guided image synthesis results on AFHQ. All images except the sources and references are generated by our proposed model, StarGAN v2. High-level semantics such as hair are followed from the references, while the pose of the sources are preserved.

E. Network architecture

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

Generator (Table 5)**.** For AFHQ, our generator consists of four downsampling blocks, four intermediate blocks, and four upsampling blocks, all of which inherit pre-activation residual units [13]. We use the instance normalization (IN) [45] and the adaptive instance normalization (AdaIN) [15, 22] 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. For CelebA-HQ, we increase the number of downsampling and upsampling layers by one. We also remove all shortcuts in the upsampling residual blocks and add skip connections with the adaptive wing based heatmap [46].

Mapping network (Table 6)**.** 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 [22] to the latent code, which has been observed not to increase model performance in our tasks. We also tried feature normalizations [3, 19], but this degraded performance.

Style encoder (Table 7)**.** 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 [16] to extract fine style features of a given reference image. The output dimension "D" in Table 7 is set to 64, which indicates the dimension of the style code.

Discriminator (Table 7)**.** Our discriminator is a multi-task discriminator [35], which contains multiple linear output branches [3]. The discriminator contains six pre-activation residual blocks with leaky ReLU [33]. 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 [19, 45] nor PatchGAN [20] 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 [36, 37, 39, 42].

[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 5. 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 6. 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 7. Style encoder and discriminator architectures. D and K represent the output dimension and number of domains, respectively.

3 implies the RGB channel...

Multi Layer Perceptron (MLP)

Latent Code is just a random 16 dim vector...

(D=64 for style encoder) in this case as it is same as the dim of the style code vector...

(D=1 for Discriminator)

References

[1] A. Almahairi, S. Rajeshwar, A. Sordoni, P. Bachman, and A. Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. In *ICML*, 2018. 2, 8

[2] A. Anoosheh, E. Agustsson, R. Timofte, and L. Van Gool. Combogan: Unrestrained scalability for image domain translation. In *CVPRW*, 2018. 2

[3] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. In *arXiv preprint*, 2016. 12

[4] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 8

[5] H. Chang, J. Lu, F. Yu, and A. Finkelstein. Pairedcyclegan: Asymmetric style transfer for applying and removing makeup. In *CVPR*, 2018. 8

[6] W. Cho, S. Choi, D. K. Park, I. Shin, and J. Choo. Image-to-image translation via group-wise deep whitening-and-coloring transformation. In *CVPR*, 2019. 8

[7] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 2, 3, 4

[8] J. Donahue and K. Simonyan. Large scale adversarial representation learning. In *NeurIPS*, 2019. 8

[9] V. Dumoulin, E. Perez, N. Schucher, F. Strub, H. d. Vries, A. Courville, and Y. Bengio. Feature-wise transformations. In *Distill*, 2018. 4

[10] 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. 8, 9

[11] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *NeurIPS*, 2017. 4

[12] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 9

[13] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 12

[14] 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. 4, 9

[15] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 2, 4, 12

[16] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 2, 3, 4, 6, 7, 8, 12

[17] L. Hui, X. Li, J. Chen, H. He, and J. Yang. Unsupervised multi-domain image translation with domain-specific encoders/decoders. In *ICPR*, 2018. 2

[18] K. Hyunsu, J. Ho Young, P. Eunhyeok, and Y. Sungjoo. Tag2pix: Line art colorization using text tag with secat and changing loss. In *ICCV*, 2019. 8

[19] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 12

[20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial nets. In *CVPR*, 2017. 1, 8, 12

[21] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 4, 9

[22] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 2, 8, 12

[23] H. Kim, M. Kim, D. Seo, J. Kim, H. Park, S. Park, H. Jo, K. Kim, Y. Yang, Y. Kim, et al. Nsml: Meet the mlaas platform with a real-world case study. *arXiv preprint arXiv:1810.09957*, 2018. 8

[24] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. In *ICML*, 2017. 3

[25] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 9

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 9

[27] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 8

[28] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. K. Singh, and M.-H. Yang. Diverse image-to-image translation via disentangled representations. In *ECCV*, 2018. 2, 4, 6, 7, 8

[29] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *NeurIPS*, 2017. 8

[30] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz. Few-shot unsupervised image-to-image translation. In *ICCV*, 2019. 2, 4, 8

[31] M. Lucic, M. Tschannen, M. Ritter, X. Zhai, O. Bachem, and S. Gelly. High-fidelity image generation with fewer labels. In *ICML*, 2019. 8

[32] L. Ma, X. Jia, S. Georgoulis, T. Tuytelaars, and L. Van Gool. Exemplar guided unsupervised image-to-image translation with semantic consistency. In *ICLR*, 2019. 8

[33] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 12

[34] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang. Mode seeking generative adversarial networks for diverse image synthesis. In *CVPR*, 2019. 2, 3, 4, 6, 7, 8

[35] L. Mescheder, S. Nowozin, and A. Geiger. Which training methods for gans do actually converge? In *ICML*, 2018. 2, 4, 8, 9, 12

[36] M. Mirza and S. Osindero. Conditional generative adversarial nets. In *arXiv preprint*, 2014. 4, 12

[37] T. Miyato and M. Koyama. cGANs with projection discriminator. In *ICLR*, 2018. 12

[38] S. Na, S. Yoo, and J. Choo. Miso: Mutual information loss with stochastic style representations for multimodal image-to-image translation. In *arXiv preprint*, 2019. 2

[39] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, 2017. 4, 12

[40] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 8

[41] 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. 9

[42] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *ICML*, 2016. 12

[43] N. Sung, M. Kim, H. Jo, Y. Yang, J. Kim, L. Lausen, Y. Kim, G. Lee, D. Kwak, J.-W. Ha, et al. Nsml: A machine learning platform that enables you to focus on your models. *arXiv preprint arXiv:1712.05902*, 2017. 8

[44] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 9

[45] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. In *arXiv preprint*, 2016. 12

[46] X. Wang, L. Bo, and L. Fuxin. Adaptive wing loss for robust face alignment via heatmap regression. In *ICCV*, 2019. 12

[47] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCV*, 2018. 8

[48] D. Yang, S. Hong, Y. Jang, T. Zhao, and H. Lee. Diversity-sensitive conditional generative adversarial networks. In *ICLR*, 2019. 3, 8

[49] 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. 9

[50] S. Yoo, H. Bahng, S. Chung, J. Lee, J. Chang, and J. Choo. Coloring with limited data: Few-shot colorization via memory augmented networks. In *CVPR*, 2019. 8

[51] X. Yu, Y. Chen, T. Li, S. Liu, and G. Li. Multi-mapping image-to-image translation via learning disentanglement. In *NeurIPS*, 2019. 8

[52] 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. 4, 9

[53] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *ICCV*, 2017. 3, 8

[54] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In *NeurIPS*, 2017. 2, 3, 4, 8