DynaGAN: Dynamic Few-shot Adaptation of GANs to Multiple Domains

Supplemental Document

Seongtae Kim
POSTECH
South Korea
seongtae0205@postech.ac.kr

Kyoungkook Kang
POSTECH
South Korea
kkang831@postech.ac.kr

Geonung Kim
POSTECH
South Korea
k2woong92@postech.ac.kr

Seung-Hwan Baek
POSTECH
South Korea
shwbaek@postech.ac.kr

Sunghyun Cho
POSTECH
South Korea
Pebblous
South Korea
s.cho@postech.ac.kr

ACM Reference Format:
Seongtae Kim, Kyoungkook Kang, Geonung Kim, Seung-Hwan Baek, and Sunghyun Cho. 2022. DynaGAN: Dynamic Few-shot Adaptation of GANs to Multiple Domains: Supplemental Document. In *SIGGRAPH Asia 2022 Conference Papers (SA '22 Conference Papers), December 6–9, 2022, Daegu, Republic of Korea.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3550469.3555416

In this Supplemental Document, we present:

- details of the style mixing,
- details of the MTG loss and identity loss,
- configurations of the experimental datasets,
- implementation details of previous methods,
- additional qualitative comparisons with previous methods,
- comparison with other weight prediction methods,
- additional qualitative comparison with multiple separate models,
- effect of the number of target domains,
- analysis of computational resources,
- results using multiple samples per domain, and
- additional examples.

S.1 Style Mixing

In order to more faithfully reflect the texture of a target domain, we adopt the style mixing strategy of MTG [Zhu et al. 2022]. Let I_c be an image of a target domain. We encode the style information of I_c to the latent code w_c by GAN inversion [Zhu et al. 2020]. At the inference time, we feed the latent code of the target domain image w_c instead of the latent code of the generator w to fine-scale layers (e.g. top 10 layers) of the synthesis network. Through the style mixing strategy, we can achieve more faithful adaptation and control the degree of adaptation as shown in Fig. 9 in the main paper.

Figure 1: Effect of the identity loss. Adapting to (b) a target domain with a large shape difference may lead to images with different identities as shown in (c). The identity loss helps preserve the source identity as shown in (d). Target: ARC Collection, Ritsumeikan University (mai04c06(2),arcBK01-0042_37).

S.2 MTG Loss and Identity Loss

For successful domain adaptation, we adopt an MTG loss $\mathcal{L}_{\text{MTG}}$ [Zhu et al. 2022] and an identity loss $\mathcal{L}_{\text{ID}}$ [Deng et al. 2019]. In this section, we describe them from the perspective of our framework. Let w^{sample} be the sampled latent code, c be the target domain that we want to synthesize, I_c be the image of the target domain c in the training dataset, and w_c be the latent code of I_c estimated by GAN inversion [Zhu et al. 2020]. Also, θ and $\hat{\theta}_c$ denote the generator's original and modulated parameters by our adaptation module, respectively.

MTG loss. The MTG loss is a weighted sum of several loss terms:

$$\begin{aligned}\mathcal{L}_{\text{MTG}} \quad = \quad & \lambda_{\text{rec}}\mathcal{L}_{\text{rec}} + \lambda_{\text{clip\_rec}}\mathcal{L}_{\text{clip\_rec}} \\ & + \lambda_{\text{domain\_gap}}\mathcal{L}_{\text{domain\_gap}} + \lambda_{\text{consist}}\mathcal{L}_{\text{consist}}\end{aligned} \tag{1}$$

where λ_{rec}, $\lambda_{\text{clip\_rec}}$, $\lambda_{\text{domain\_gap}}$ and λ_{consist} are balancing weights. We set $\lambda_{\text{rec}} = 10$, $\lambda_{\text{clip\_rec}} = 30$, $\lambda_{\text{domain\_gap}} = 1$, and $\lambda_{\text{consist}} = 0.5$.

SA '22 Conference Papers, December 6–9, 2022, Daegu, Republic of Korea

ACM ISBN 978-1-4503-9470-3/22/12…$15.00
https://doi.org/10.1145/3550469.3555416

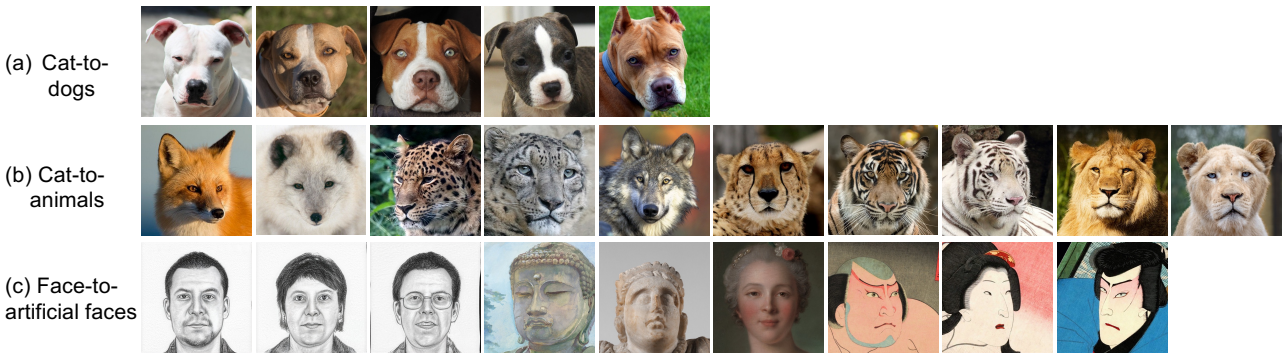

Figure 2: Experimental datasets. (a) The cat-to-dogs dataset consists of five target images sampled from the AFHQ Dog dataset [Choi et al. 2020]. (b) The cat-to-animals dataset consists of 10 target images of different animal species from the AFHQ Wild dataset [Choi et al. 2020]. (c) The real-to-artificial faces dataset consists of nine target images (three sketch images synthesized by FS-ada using the Sketch dataset [Ojha et al. 2021], three images from the MetFace dataset [Karras et al. 2020], and three images from the Ukiyo-e dataset [Pinkney 2020]). Images in (a) and (b) from left to right: Pixabay (rmacwheeler, sandid, Doz777, TC-TORRES, vilinapetrova, Pexels, dclobes, vinzling, Pixel-mixer, WikiImages, jmrockeman, Gregorius_o, JayDLim, DonStott, PhotoGranary) [Pixabay License]. 4th, 5th and 6th images in (c): The Metropolitan Museum of Art [Public Domain]. 7th, 8th and 9th images in (c): ARC Collection, Ritsumeikan University (mai04c06(2),arcBK01-0042_37,shiUYa0007).

First, DynaGAN should take w_c and c as input, and reconstruct the target image I_c where I_c is an image in the training set, and w_c is its GAN latent code found by an off-the-shelf GAN inversion method [Tov et al. 2021]. To this end, a reconstruction loss $\mathcal{L}_{\text{rec}}$ is defined as:

$$\mathcal{L}_{\text{rec}} = \mathcal{L}_2(I_c, G(w_c; \hat{\theta}_c)) + \mathcal{L}_{\text{LPIPS}}(I_c, G(w_c; \hat{\theta}_c)) \tag{2}$$

where $\mathcal{L}_2$ and $\mathcal{L}_{\text{LPIPS}}$ are a pixel-wise MSE loss, and a perceptual loss [Zhang et al. 2018], respectively.

Second, the reconstructed image should be semantically close to the target image I_c. We evaluate the semantic similarity based on the cosine similarity in the CLIP embedding space. A semantic reconstruction loss $\mathcal{L}_{\text{clip\_rec}}$ is defined:

$$\mathcal{L}_{\text{clip\_rec}} = 1 - \text{sim}\left(E_{\text{CLIP}}(I_c), E_{\text{CLIP}}(G(w_c; \hat{\theta}_c))\right) \tag{3}$$

where, $\text{sim}(\cdot)$ is cosine similarity and E_{CLIP} is a CLIP encoder [Radford et al. 2021].

Third, the domain gap between the source domain and target domain should be maintained not only for the latent code w_c but also for any arbitrary latent code w^{sample}. Specifically, a domain adaptation direction vector is defined in the CLIP embedding space as: $v = E_I(I_c) - E_I(G(w_c; \theta))$. Likewise, another domain adaptation direction vector for a sampled latent code w^{sample} is defined as: $v^{\text{sample}} = E_I(G(w^{\text{sample}}; \hat{\theta}_c)) - E_I(G(w^{\text{sample}}; \theta))$. To enforce these two directions to be similar, a domain gap loss $\mathcal{L}_{\text{domain\_gap}}$ is defined as:

$$\mathcal{L}_{\text{domain\_gap}} = 1 - \text{sim}(v, v^{\text{sample}}). \tag{4}$$

Lastly, the semantic difference between w_c and w^{sample} in the source domain should be maintained in the target domain as well. For this purpose, two directional vectors are defined: $v_s = E_I(G(w_c; \theta)) - E_I(G(w^{\text{sample}}; \theta))$ and $v_c = E_I(I_c) - E_I(G(w^{\text{sample}}; \hat{\theta}_c))$, and a domain consistency loss $\mathcal{L}_{\text{consist}}$ is defined:

$$\mathcal{L}_{\text{consist}} = 1 - \text{sim}(v_c, v_s) \tag{5}$$

Identity loss. In the case of adapting to a dataset composed of human portrait images, we adopt the identity loss $\mathcal{L}_{\text{ID}}$ [Deng et al. 2019] to encourage a synthesized image to be similar to a source-domain image in the representation space of a face recognition network. $\mathcal{L}_{\text{ID}}$ is defined as:

$$\mathcal{L}_{\text{ID}} = 1 - \text{sim}\left(\text{Arc}(G(w^{\text{sample}}; \theta)), \text{Arc}(G(w^{\text{sample}}; \hat{\theta}_c))\right) \tag{6}$$

where $\text{Arc}(\cdot)$ is a pretrained ArcFace network [Deng et al. 2019] that extracts the representation of each portrait image. Fig. 1 shows the effect of the identity loss. As shown in the figure, the identity loss helps preserve the identity of the source image after adaptation.

S.3 Comparisons with Other Weight Prediction Methods

We compare our efficient weight modulation method with two other methods proposed by HyperStyle [Alaluf et al. 2022], which are the per-channel prediction and separable kernel prediction methods. The per-channel prediction method estimates a channel-wise tensor of size $C_{\text{out}} \times C_{\text{in}} \times 1$ and replicates it in a spatial kernel footprint. On the other hand, the separable kernel prediction method estimates a weight offset by the product of two decomposed tensors of sizes $C_{\text{out}} \times 1 \times k$ and $1 \times C_{\text{in}} \times k$. The per-channel prediction and separable kernel prediction methods estimate 262,144 and 9,216 parameters, respectively. Meanwhile, DynaGAN predicts a filter-wise scaling parameter δ and rank-1 decomposed tensors γ, ϕ, and ψ. While DynaGAN estimates the fewest parameters (e.g. 1,545) for the convolutional layers, it produces comparable results to the others as shown in Fig. 3.

S.4 Large Number of Target Domains

We show the effect of the number of target domains. We curate three training datasets with different numbers of domains. Each dataset consists of 10, 30, and 50 different caricature styles sampled from StyleCariGAN [Jang et al. 2021]. As shown in Fig. 4, the quality

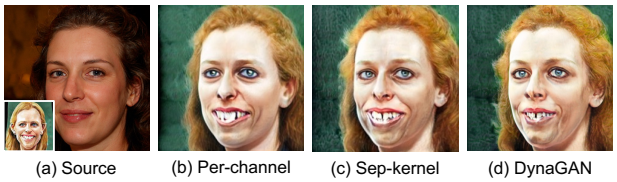

Figure 3: Qualitative comparison among different parameter prediction methods: (b) Per channel prediction, (c) Separable kernel prediction, and (d) DynaGAN. The inset image in (a) is the target caricature image. DynaGAN achieves successful domain adaptation even with the fewest parameters.

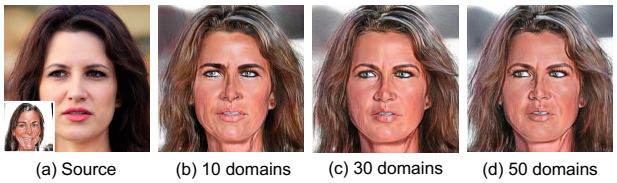

Figure 4: Effect of the number of target domains. (a) is a source domain image and the inset is a target domain image. (b), (c), and (d) are samples of DynaGAN trained with 10, 30 and 50 target domains, respectively. The quality of the results does not degrade as the number of target domains increases, which indicates that our adaptation module can handle a large number of target domains as well.

of the results does not degrade as the number of target domains increases, which indicates that our adaptation module can handle a large number of target domains as well.

S.5 Experimental Datasets

Here, we display all the experimental datasets used for the comparisons with previous methods in Sec. 4.2 in the main paper in Fig. 2 and describe their detailed configuration. The cat-to-dogs dataset consists of five dog face images of the same breed (American Staffordshire Terrier) selected from the AFHQ Dog dataset [Choi et al. 2020]. For quantitative comparison in the main paper, we used 171 American Staffordshire Terrier images from the AFHQ Dog dataset. For training DynaGAN, we set the number of iterations to 1,200.

The cat-to-animals dataset consists of 10 animal face images of different species (fox, snow fox, cheetah, white leopard, wolf, leopard, tiger, white tiger, male lion, and female lion) from the AFHQ Wild dataset [Choi et al. 2020]. For quantitative comparison, we used 483 animal face images of the validation dataset of the AFHQ Wild dataset. For training DynaGAN, we set the number of iterations to 1,200.

The real-to-artificial faces dataset consists of nine human face images with different styles. Specifically, we obtain three sketch images using FS-ada from the Sketch dataset [Ojha et al. 2021], three images from the MetFace dataset [Karras et al. 2020], and three images from the Ukiyo-e dataset [Pinkney 2020]. For quantitative comparison, we used 900 human face images, selected from each dataset with the same number. For training DynaGAN, we set the number of iterations to 600.

S.6 Implementation Details of Previous Methods

TGAN [Wang et al. 2018] & TGAN-ada [Karras et al. 2020]. We used an unofficial code[1] for TGAN [Wang et al. 2018] and TGAN-ada [Karras et al. 2020] with default settings. We set the batch size to 4 and the number of iterations to 10,000.

FS-ada. We used the authors code[2] for FS-ada [Ojha et al. 2021] with default settings. We set the batch size to 4. We set the number of iterations to 10,000 and 5,000 for cat-to-dogs and cat-to-animals, and real-to-artificial faces, respectively.

StyleGAN-nada. We used the author's code[3] for StyleGAN-nada [Gal et al. 2022] with default settings. We set the number of iterations to 2,000 and 300 for cat-to-dogs and cat-to-animals, and real-to-artificial faces, respectively.

MTG-ext. Since MTG [Zhu et al. 2022] aims at single-shot domain adaptation, we extend it to MTG-ext for the few-shot adaptation task. Specifically, MTG-ext evaluates the MTG loss on each target image and averages them. At the inference time, MTG-ext also applies the style mixing strategy of MTG [Zhu et al. 2022]. In our implementation, we used the author's code[4] for MTG-ext with default settings. We set the number of iterations to 1,200 and 600 for cat-to-dogs and cat-to-animals, and real-to-artificial faces, respectively.

S.7 Additional Qualitative Comparisons with Previous Methods

Figs. 5, 6, and 7 show additional qualitative comparisons of different methods on each experimental dataset including the results of all of the training images. As also shown in Fig. 4 in the main paper, our method successfully produces domain adaptation results compared to the previous methods.

S.8 Additional Qualitative Comparison with Multiple Separate Models

One naïve multi-domain adaptation solution is to train multiple separate models for each target domain. We show additional qualitative comparison results between this naïve approach and DynaGAN in Fig. 8. As DynaGAN learns complemental characteristics from different target domains and efficiently modulates the parameters, DynaGAN results in more faithful multi-domain adaptation than multiple FS-ada [Ojha et al. 2021], StyleGAN-nada [Gal et al. 2022], and MTG [Zhu et al. 2022] models.

S.9 Computational Resources

In this section, we analyze the computational requirements for our method. First, we report the number of parameters to be estimated by the adaptation module according to how to change the weight of

[1] https://github.com/rosinality/stylegan2-pytorch
[2] https://github.com/utkarshojha/few-shot-gan-adaptation
[3] https://github.com/rinongal/StyleGAN-nada
[4] https://github.com/ZPdesu/MindTheGap

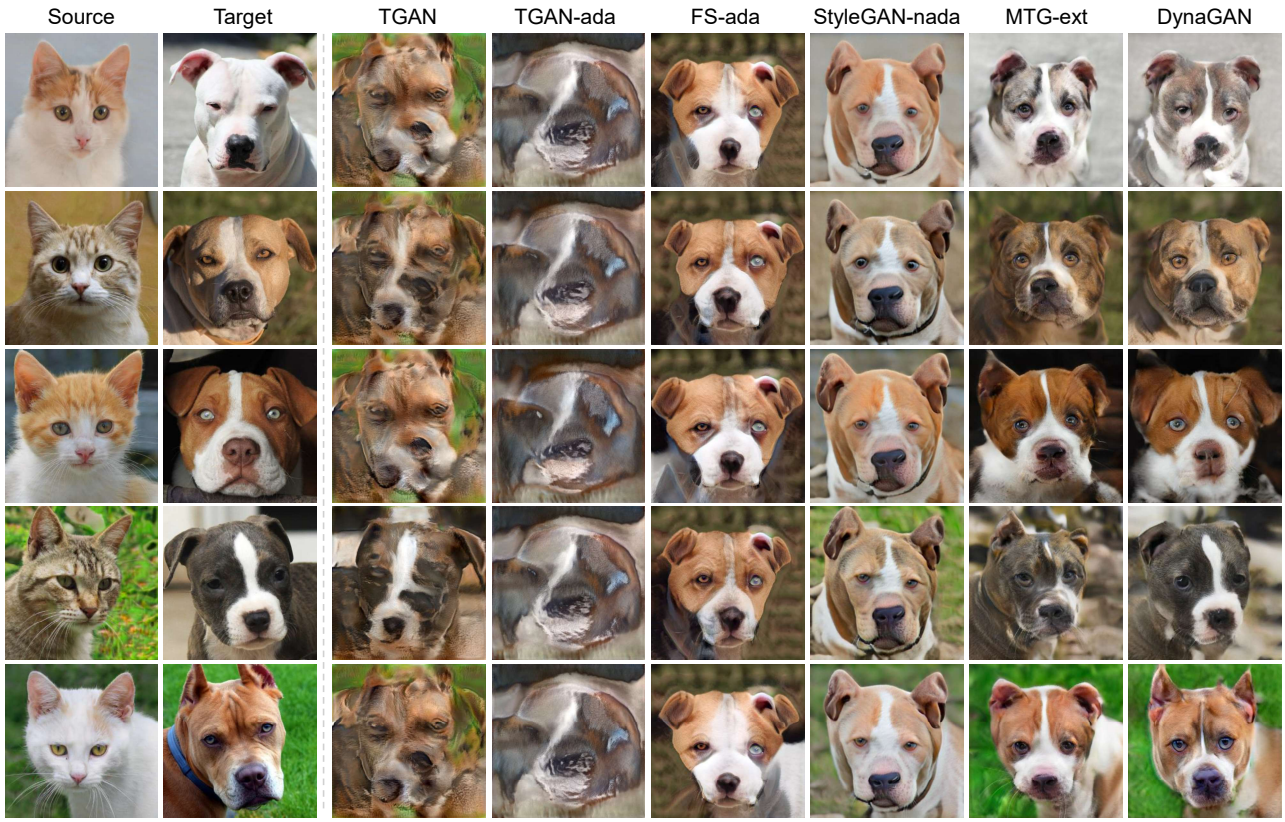

Figure 5: Qualitative comparison of different methods on the cat-to-dogs dataset. Targets from top to bottom: Pixabay (rmacwheeler, sandid, Doz777, TC-TORRES, vilinapetrova) [Pixabay License].

the convolutional layer of the generator, when we train DynaGAN on images of size 1024×1024. Estimating the residual parameters of all weights requires 12.1B parameters, which requires unacceptably large memory and training time. Also, estimating channel-wise residuals suggested by HyperStyle [Alaluf et al. 2022] requires about 1.3B parameters to be estimated. On the other hand, our approach requires only 9.4M parameters thanks to filter-wise scaling for δ and rank-1 tensor decomposition for $\Delta\varphi$.

Here, we also report the training time of DynaGAN, which is measured on a single NVIDIA RTX A6000 GPU. DynaGAN takes about 26 minutes to adapt a generator to the real-to-artificial faces dataset, which is much shorter than 99 minutes of multi-MTGs, and this gap gets gradually bigger as the number of target domains increases.

S.10 Multiple Samples per Domain

Although DynaGAN aims at extreme few-shot adaptation with one image per domain, it is easily scalable to multiple images per domain. In our implementation, we can modify l_{pos} of Eq. 6 in the main paper as:

$$l_{\text{pos}} = \text{sim}\left(\text{avg}\left(E_{\text{CLIP}}(I_c)\right), E_{\text{CLIP}}(\hat{I}_c(w))\right) \tag{7}$$

where we use an average of training images of a target domain c instead of a single image I_c. Fig. 9 compares the results of DynaGAN trained on the cat-to-animals dataset consisting of one image per domain with the results of DynaGAN trained on images consisting of three images per domain. As shown in Fig. 9, multiple images help our model better model each domain, as it can use domain information shared across multiple images.

S.11 Additional Examples

Domain interpolation. DynaGAN supports smooth domain interpolation as discussed in the main paper. Fig. 10 shows additional domain interpolation examples between two target domains.

Additional examples. We present additional examples of DynaGAN in Figs. 11, 12, and 13 showing domain adaptation results of various source domain images. In these examples, we use the FFHQ dataset [Karras et al. 2019] for the source domains, and compare the results of TGAN [Wang et al. 2018], TGAN-ada [Karras et al. 2020], FS-ada [Ojha et al. 2021], StyleGAN-nada [Gal et al. 2022], MTG-ext [Zhu et al. 2022], and ours. Although we use only one image per target domain, DynaGAN achieves successful adaptation results that faithfully reflect target-domain characteristics.

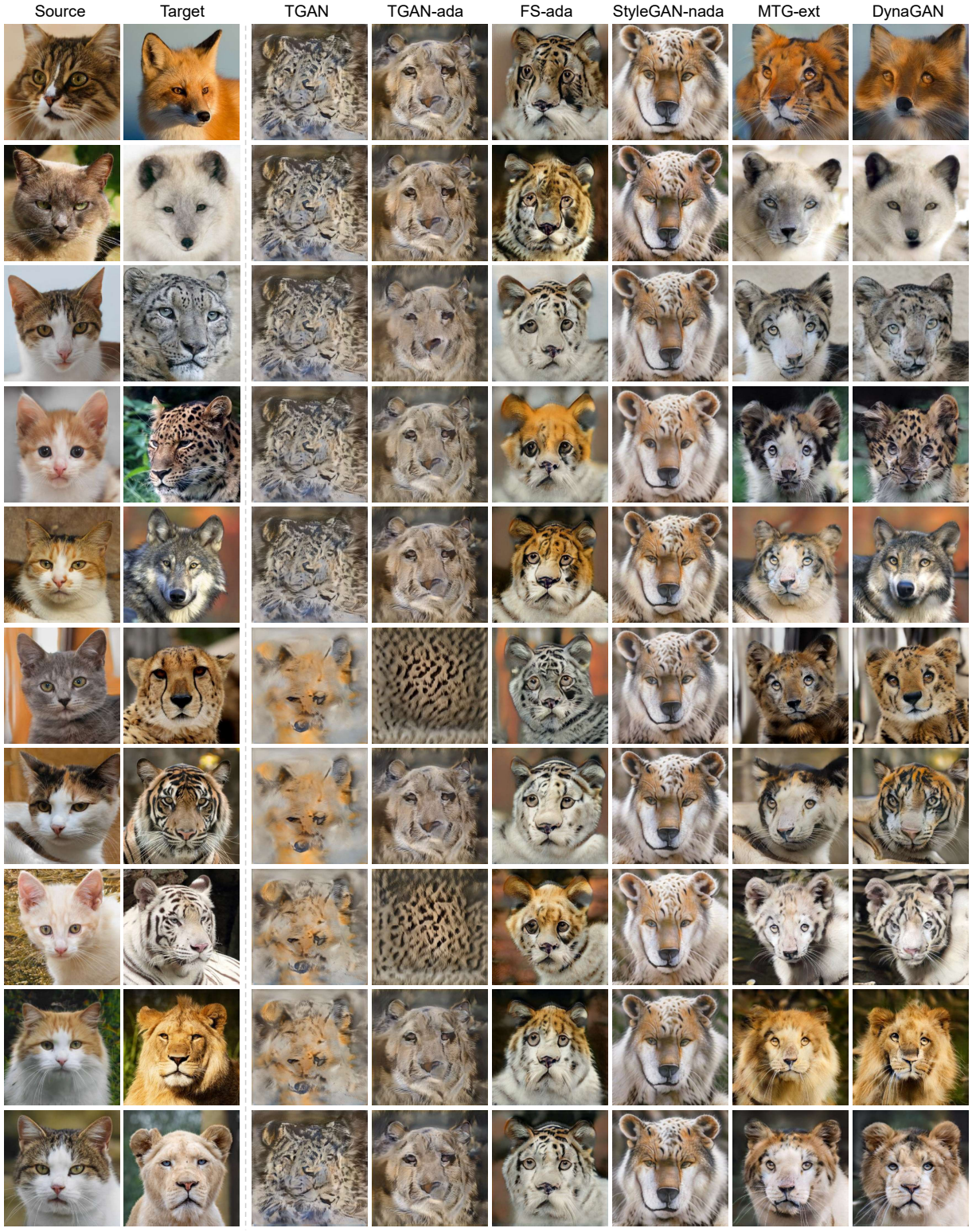

Figure 6: Qualitative comparison of different methods on the cat-to-animals dataset. Targets from top to bottom: Pixabay (Pexels, dclobes, vinzling, Pixel-mixer, WikiImages, jmrockeman, Gregorius_o, JayDLim, DonStott, PhotoGranary) [Pixabay License].

Figure 7: Qualitative comparison of different methods on the real-to-artificial faces dataset. 4th, 5th and 6th target images: The Metropolitan Museum of Art [Public Domain]. 7th, 8th and 9th target images: ARC Collection, Ritsumeikan University (mai04c06(2),arcBK01-0042_37,shiUYa0007).

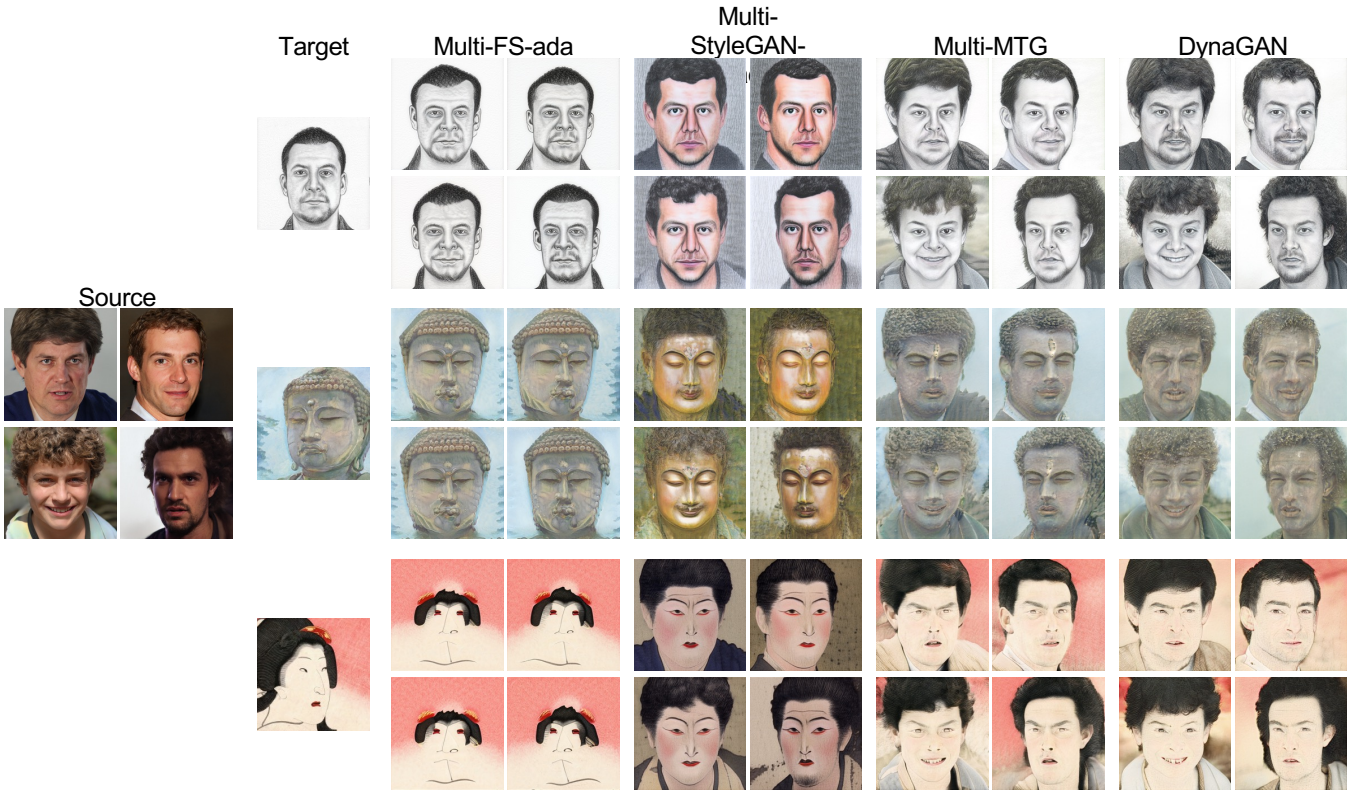

Figure 8: Qualitative comparison with multiple separate models on the real-to-artificial faces dataset. Multi-FS-ada [Ojha et al. 2021], Mult-StyleGAN-nada [Gal et al. 2022], and Multi-MTG [Zhu et al. 2022] models are trained separately for each target domain. Thus, their results are overfitted to each target domain. On the other hand, our method performs better domain adaptation while preventing overfitting with only a single model. 2nd target image: The Metropolitan Museum of Art [Public Domain]. 3rd target image: ARC Collection, Ritsumeikan University (mai04c06(2)).

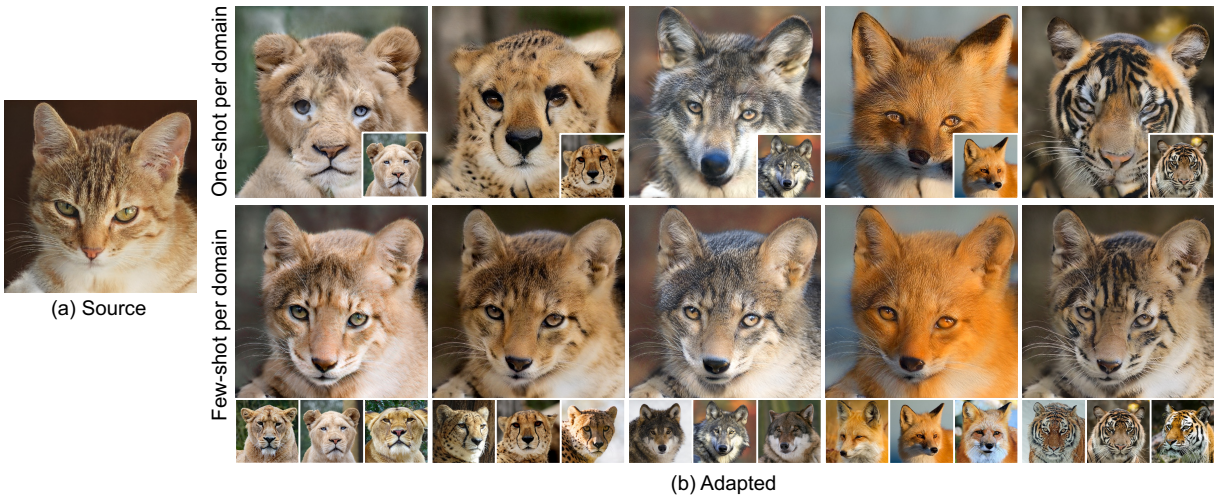

Figure 9: Qualitative comparison of training our method on one-shot / few-shot images per domain. The insets at the bottom of the figure are training data for each domain. Insets: Pixabay (PhotoGranary, jmrockeman, WikiImages, Pexels, Gregorius_o, Kevinsphotos, bbuchmayr0, AnnyksPhotography, luxstorm, Wilda3, 942784, 12019, Ralphs_Fotos, Pixel-mixer, blende12) [Pixabay License].

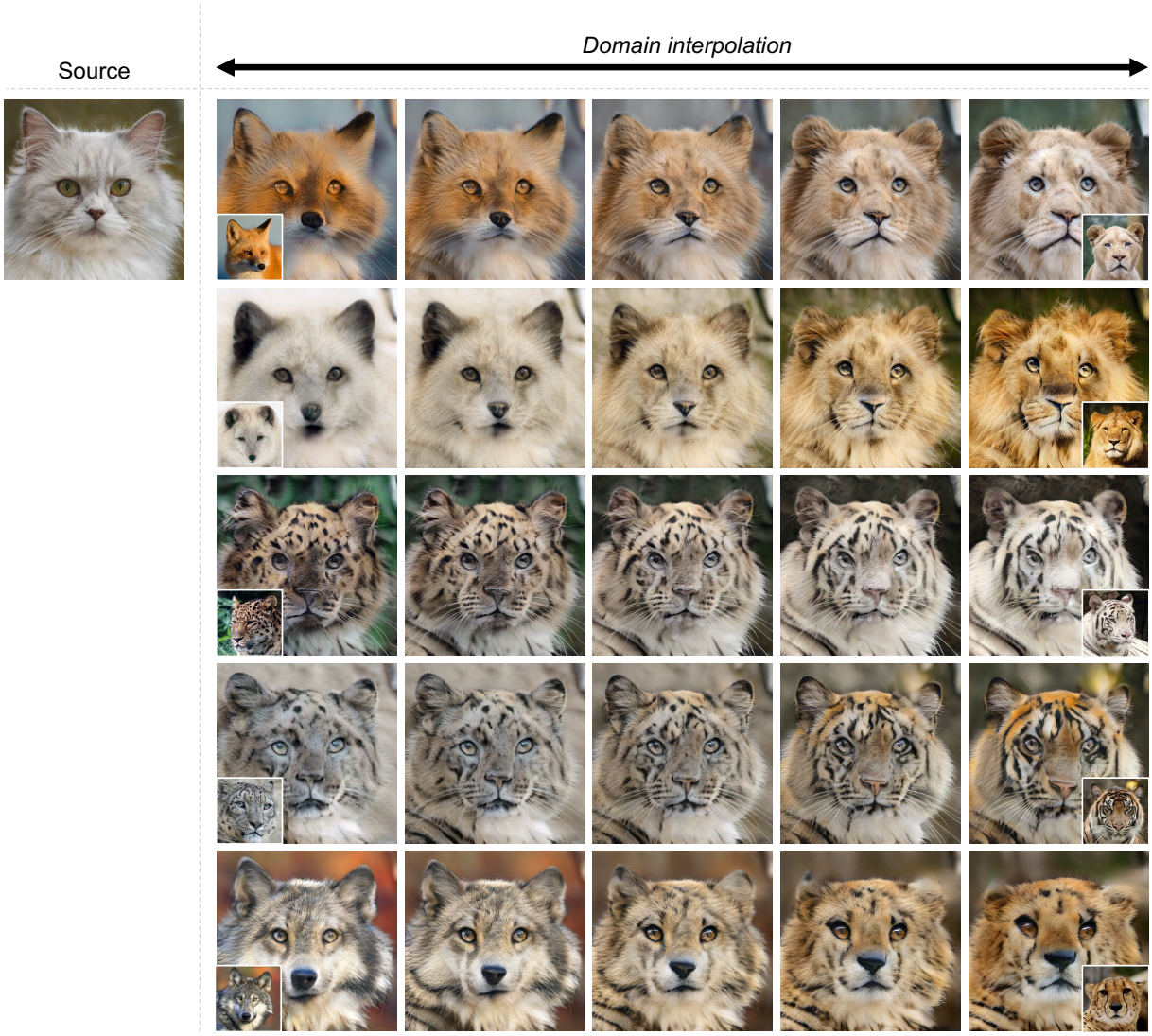

Figure 10: Additional domain interpolation results between target domains of the cat-to-animals dataset. The insets are target domains. DynaGAN can synthesize new domains by interpolating the target-domain condition vector c. Insets from left to right, top to bottom: Pixabay (Pexels, PhotoGranary, dclobes, DonStott, vinzling, JayDLim, Pixel-mixer, Gregorius_o, WikiImages, jmrockeman) [Pixabay License].

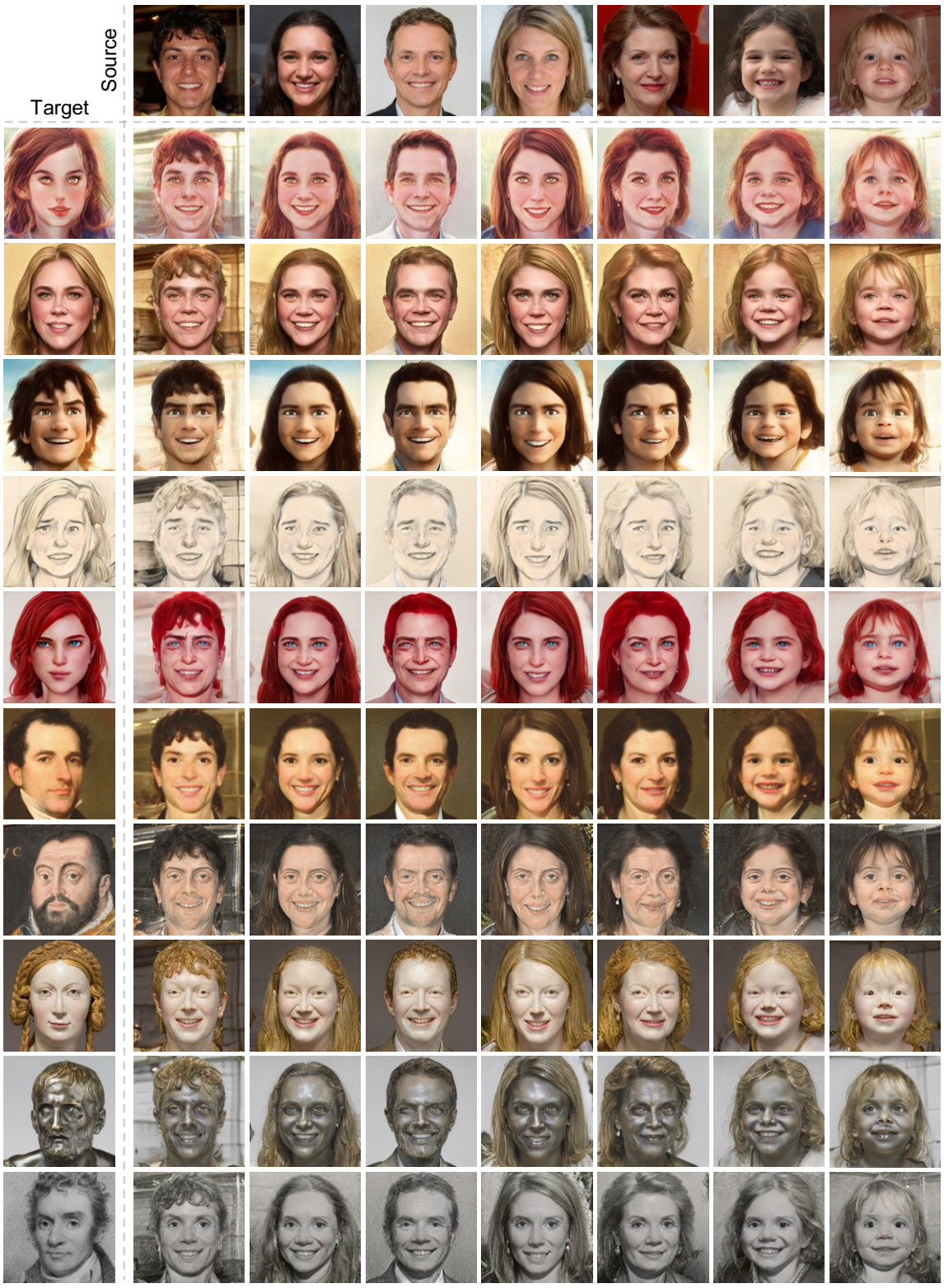

Figure 11: Additional results of DynaGAN. Bottom five target images: The Metropolitan Museum of Art [Public Domain].

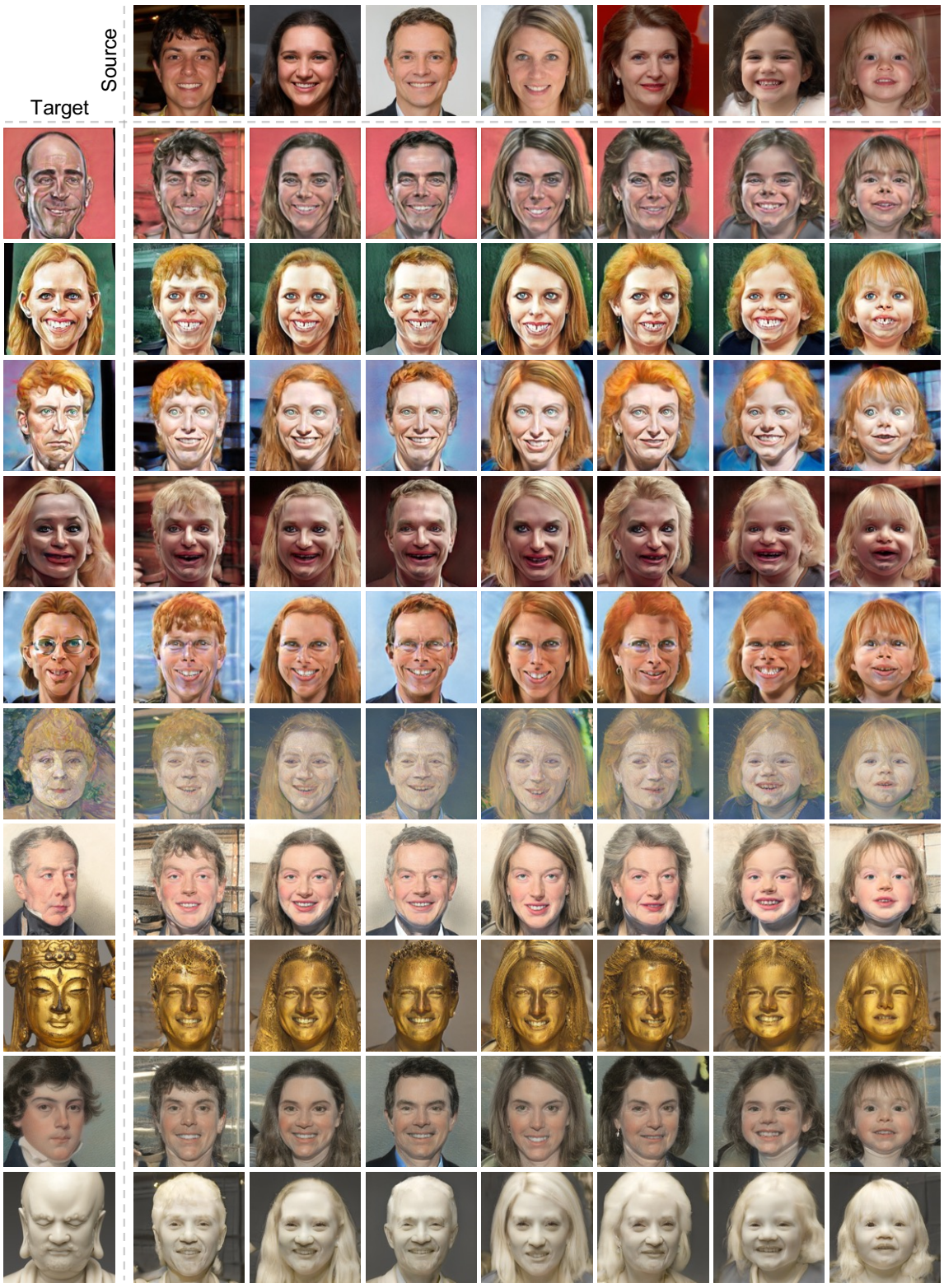

Figure 12: Additional results of DynaGAN. Bottom five target images: The Metropolitan Museum of Art [Public Domain].

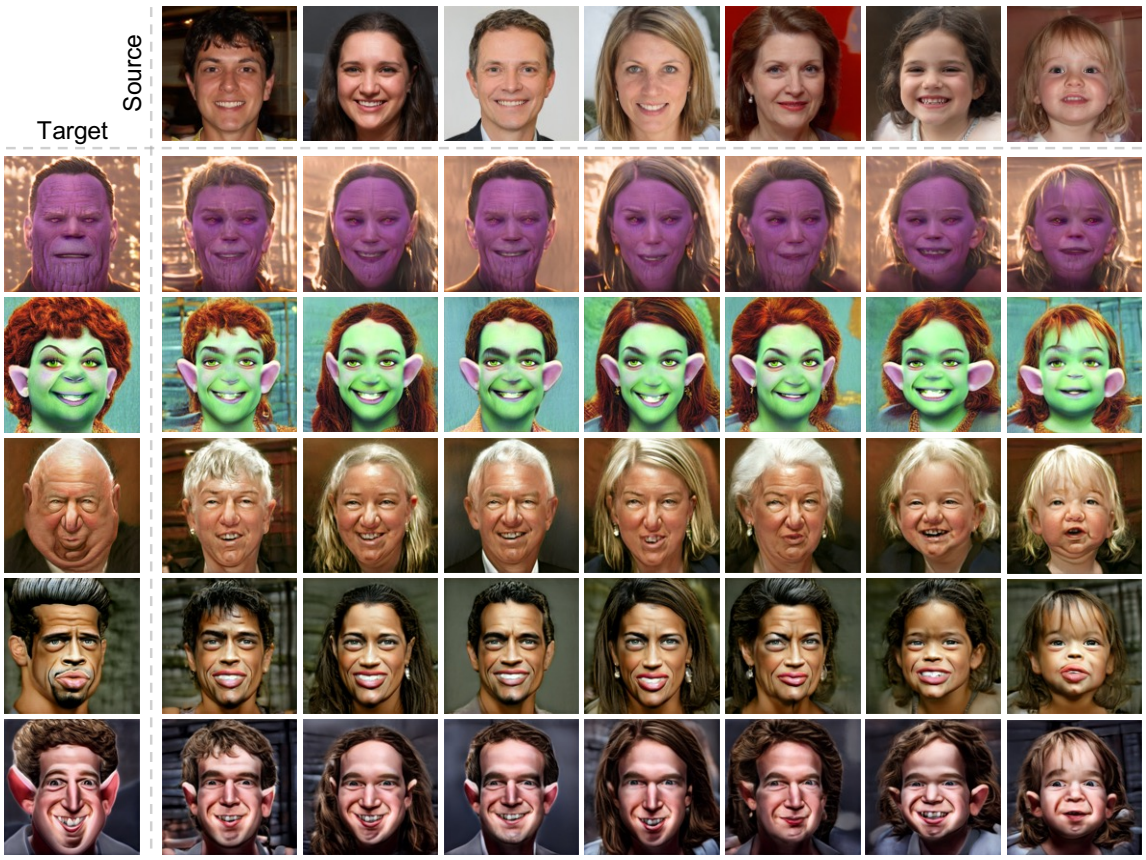

Figure 13: Additional results of DynaGAN.

REFERENCES

Yuval Alaluf, Omer Tov, Ron Mokady, Rinon Gal, and Amit Bermano. 2022. Hyperstyle: Stylegan inversion with hypernetworks for real image editing. In *Proc. of IEEE/CVF CVPR*.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. StarGAN v2: Diverse Image Synthesis for Multiple Domains. In *Proc. of IEEE/CVF CVPR*.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. ArcFace: Additive angular margin loss for deep face recognition. In *Proc. of IEEE/CVF CVPR*.

Rinon Gal, Or Patashnik, Haggai Maron, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. 2022. StyleGAN-NADA: CLIP-Guided Domain Adaptation of Image Generators. *ACM Trans. Graph.* 41, 4, Article 141 (jul 2022), 13 pages. https://doi.org/10.1145/3528223.3530164

Wonjong Jang, Gwangjin Ju, Yucheol Jung, Jiaolong Yang, Xin Tong, and Seungyong Lee. 2021. StyleCariGAN: Caricature Generation via StyleGAN Feature Map Modulation. *ACM Trans. Graph.* 40, 4, Article 116 (jul 2021), 16 pages. https://doi.org/10.1145/3450626.3459860

Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. 2020. Training generative adversarial networks with limited data. In *Proc. of NeurIPS*.

Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proc. of IEEE/CVF CVPR*.

Utkarsh Ojha, Yijun Li, Jingwan Lu, Alexei A Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. 2021. Few-shot image generation via cross-domain correspondence. In *Proc. of IEEE/CVF CVPR*.

Justin N. M. Pinkney. 2020. Aligned Ukiyo-e faces dataset. https://www.justinpinkney.com/ukiyoe-dataset.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proc. of ICML*.

Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. 2021. Designing an Encoder for StyleGAN Image Manipulation. *ACM Trans. Graph.* 40, 4, Article 133 (jul 2021), 14 pages. https://doi.org/10.1145/3450626.3459838

Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, and Bogdan Raducanu. 2018. Transferring GANs: generating images from limited data. In *Proc. of ECCV*.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. of IEEE/CVF CVPR*.

Peihao Zhu, Rameen Abdal, John Femiani, and Peter Wonka. 2022. Mind the Gap: Domain Gap Control for Single Shot Domain Adaptation for Generative Adversarial Networks. In *Proc. of ICLR*.

Peihao Zhu, Rameen Abdal, Yipeng Qin, John Femiani, and Peter Wonka. 2020. Improved stylegan embedding: Where are the good latents? *arXiv preprint arXiv:2012.09036* (2020).