Appendix: Rethinking the Truly Unsupervised Image-to-Image Translation

1. Qualitative comparison on the number of pseudo domains $\hat{K}$

Please refer to Figure 1. in Appendix.

2. Training details

We train the guiding network for the first 65K iterations while freezing the update from both the generator and the discriminator. Then, we train the whole framework 100K more iterations for training all the networks. The batch size is set to 32 and 16 for 128×128 and 256×256 images, respectively. Training takes about 36 hours on a single Tesla V100 GPU with our implementation using PyTorch[11]. We use Adam [5] optimizer with $\beta_1 = 0.9, \beta_2 = 0.99$ for the guiding network, and RMSprop [3] optimizer with $\alpha = 0.99$ for the generator and the discriminator. All learning rates are set to 0.0001 with a weight decay 0.0001. We adopt hinge version adversarial loss [6, 13] with R_1 regularization [8] using $\gamma = 10$ (Eq. 5). We set $\lambda_{\text{rec}} = 0.1, \lambda^G_{\text{style}} = 0.01, \lambda^E_{\text{style}} = 1$, and $\lambda_{\text{MI}} = 5$ in equation. 6 for all experiments. When the guiding network is simultaneously trained with the generator, we decrease λ^E_{style} and λ_{MI} to 0.1 and 0.5, respectively. For evaluation, we use the exponential moving average over the parameters [4] of the guiding network and the generator. We initialize the weights of convolution layers with He initialization [1], all biases to zero, and weights of linear layers from $N(0, 0.01)$ with zero biases. The source code will be available publicly. The source code is available at https://github.com/clovaai/tunit.

3. Evaluation protocol

For evaluation, we use class-wise Fréchet Inception Distance (FID) [2], which is often called mFID in literatures and D&C [9]. FID measures Fréchet distance between real and fake samples embedded by the last average pooling layer of Inception-V3 pre-trained on ImageNet. Class-wise FID is obtained by averaging the FIDs of individual classes. In the experiments with fewer labels, we report the mean value of best five mFID's over 100K iterations. For example, we use entire real images of each class and generate 810 fake images where $18 \times (K - 1)$ source images ($K = 10$ for AnimalFaces-10) and five reference images of AnimalFaces-10 are used to produce those fake images. We choose the source images from all classes except for the target class. For each source image, the five references are selected arbitrarily. For D&C, we generate fake images the similar number of training images with randomly selected source and reference images. Then, we use Inception-V3 pre-trained on ImageNet for extracting feature vectors and measure D&C by using the feature vectors.

4. Difference between the sequential and joint training in Section 4.1

To investigate the effect of the adversarial loss to the guiding network, we trained TUNIT under two settings; 1) joint training and 2) sequential training. The former is to train all the networks in an end-to-end manner as described in Section 3, and the latter is to first train the guiding network with $\mathcal{L}_E$ for 100k iterations and then train the generator and the discriminator using the outputs of the frozen guiding network as their inputs. Note that for the separate training, the guiding network does not receive feedback from the translation loss $\mathcal{L}_G$ in Eq. (6).

5. Architecture details

For the guiding network, we use `VGG11` before the linear layers followed by the average pooling operation as the shared part and append two branches E_{class} and E_{style}. The branches are one linear layer with $\hat{K}$ and 128 dimensional outputs, respectively. The detailed information of the generator, the guiding network and the discriminator architectures are provided in Table 1, Table 2 and Table 3.

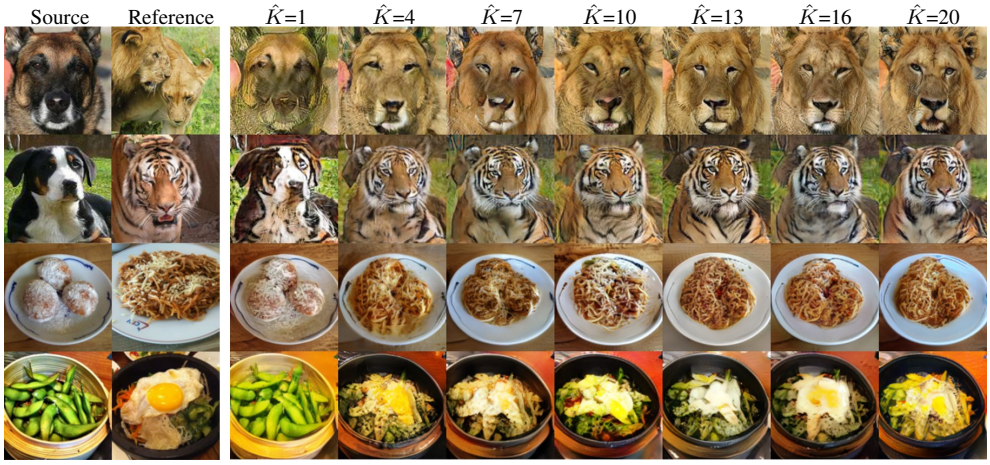


Figure 1: Qualitative comparison on the number of pseudo domains $\hat{K}$. The performance varies along with $\hat{K}$. When we set $\hat{K}$ large enough, the results are reasonable.

LAYER	RESAMPLE	NORM	OUTPUT SHAPE
Image $\mathbf{x}$	-	-	$128 \times 128 \times 3$
Conv7×7	-	IN	$128 \times 128 \times ch$
Conv4×4	Stride 2	IN	$64 \times 64 \times 2ch$
Conv4×4	Stride 2	IN	$32 \times 32 \times 4ch$
Conv4×4	Stride 2	IN	$16 \times 16 \times 8ch$
ResBlk	-	IN	$16 \times 16 \times 8ch$
ResBlk	-	IN	$16 \times 16 \times 8ch$
ResBlk	-	AdaIN	$16 \times 16 \times 8ch$
ResBlk	-	AdaIN	$16 \times 16 \times 8ch$
Conv5×5	Upsample	AdaIN	$32 \times 32 \times 4ch$
Conv5×5	Upsample	AdaIN	$64 \times 64 \times 2ch$
Conv5×5	Upsample	AdaIN	$128 \times 128 \times ch$
Conv7×7	-	-	$128 \times 128 \times 3$

Table 1: Generator architecture. "ch" represents the channel multiplier that is set to 64. IN and AdaIN indicate instance normalization and adaptive instance normalization, respectively.

LAYER	RESAMPLE	NORM	OUTPUT SHAPE
Image $\mathbf{x}$	-	-	$128 \times 128 \times 3$
Conv3×3	MaxPool	BN	$64 \times 64 \times ch$
Conv3×3	MaxPool	BN	$32 \times 32 \times 2ch$
Conv3×3	-	BN	$32 \times 32 \times 4ch$
Conv3×3	MaxPool	BN	$16 \times 16 \times 4ch$
Conv3×3	-	BN	$16 \times 16 \times 8ch$
Conv3×3	MaxPool	BN	$8 \times 8 \times 8ch$
Conv3×3	-	BN	$8 \times 8 \times 8ch$
Conv3×3	MaxPool	BN	$4 \times 4 \times 8ch$
GAP	-	-	$1 \times 1 \times 8ch$
FC	-	-	128
FC	-	-	$\hat{K}$

Table 2: Guiding network architecture. "ch" represents the channel multiplier that is set to 64. The architecture is based on VGG11-BN. GAP and FC denote global average polling [7] and fully connected layer, respectively.

LAYER	RESAMPLE	NORM	OUTPUT SHAPE
Image $\mathbf{x}$	-	-	$128 \times 128 \times 3$
Conv3×3	-	-	$128 \times 128 \times ch$
ResBlk	-	FRN	$128 \times 128 \times ch$
ResBlk	AvgPool	FRN	$64 \times 64 \times 2ch$
ResBlk	-	FRN	$64 \times 64 \times 2ch$
ResBlk	AvgPool	FRN	$32 \times 32 \times 4ch$
ResBlk	-	FRN	$32 \times 32 \times 4ch$
ResBlk	AvgPool	FRN	$16 \times 16 \times 8ch$
ResBlk	-	FRN	$16 \times 16 \times 8ch$
ResBlk	AvgPool	FRN	$8 \times 8 \times 16ch$
ResBlk	-	FRN	$8 \times 8 \times 16ch$
ResBlk	AvgPool	FRN	$4 \times 4 \times 16ch$
LReLU	-	-	$4 \times 4 \times 16ch$
Conv4×4	-	-	$1 \times 1 \times 16ch$
LReLU	-	-	$1 \times 1 \times 16ch$
Conv1×1	-	-	$\hat{K}$

Table 3: Discriminator architecture. "ch" and $\hat{K}$ represent the channel multiplier that is set to 64 and the number of clusters, respectively. FRN indicates filter response normalization [12].

6. Comparison with Swapping autoencoder

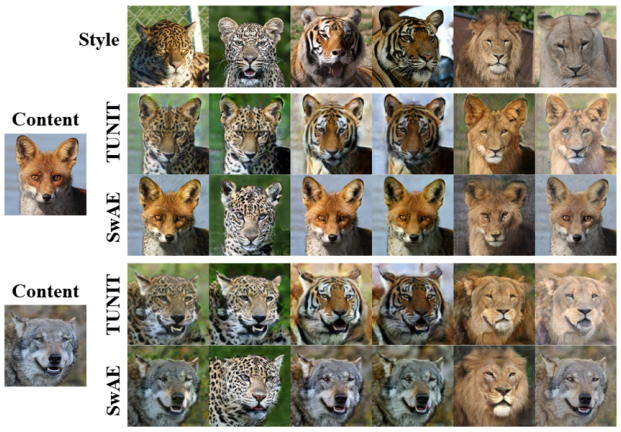


Figure 2: Comparison with SwAE. SwAE sometimes fails to capture the domain features. We recommend to zoom in.

Swapping autoencoder (SwAE) [10] can conduct the image translation without the domain labels by using the feature vector of the reference images. Therefore, it can be compared with TUNIT. Figure 2 shows the qualitative comparison between TUNIT and SwAE. Since SwAE does not define domains, it occasionally fails to capture the exact domain properties (col. 1,3,4 and 6). Meanwhile, TUNIT captures various aspects of domains; it changes species along with styles. It shows that TUNIT better handles translation across domains. Originally, we did not compare SwAE because their practical usefulness and the possible tasks differ from ours. Notably, similar to many unsupervised learners, TUNIT can serve as a strong baseline for semi-supervised models. Table 3 and Fig. 4 show that TUNIT is successful in a semi-supervised setting. This is clearly not possible by SwAE due to their design choice. Besides, TUNIT further translates domains specified by cluster ids via their average styles (Appendix Fig. 10). It is especially useful when a user wants to explore various domains without references. In contrast, SwAE always requires a reference.

7. Comparison with StarGANv2

We additionally compare TUNIT with StarGANv2 on AnimalFaces-10 and Food-10. We employ StarGANv2 with supervision as the reference for the upper-bound. The table below shows the quantitative result.

	AnimalFaces-10		**Food-10**	
	mFID	D&C	mFID	D&C
StarGANv2 (Supervised)	33.67	1.54/0.91	65.03	1.09/0.76
TUNIT (Unsupervised)	47.70	1.04/0.81	52.20	1.08/0.87

StarGANv2 outperformed unsupervised TUNIT on AnimalFaces-10, but TUNIT outperformed StarGANv2 on Food-10. Considering that TUNIT uses no labels and StarGANv2 uses set-level labels, we emphasize that our achievement is impressive.

8. Perceptual study on disentanglement

We conducted the user study (selecting the best in style and content) on two datasets and compared models (from Table 1) as follows. The result shows that the proposed (F, G) largely outperforms the others.

	A	B	C	D	E	F	G
Preference(%)↑	3.0	6.2	3.2	10.8	12.3	19.2	45.3

9. t-SNE visualization & cluster example images

Figure 3: **t-SNE visualization and representative images of each domain for AFHQ Cat.**

Figure 4: **t-SNE visualization and representative images of each domain for AFHQ Dog.**

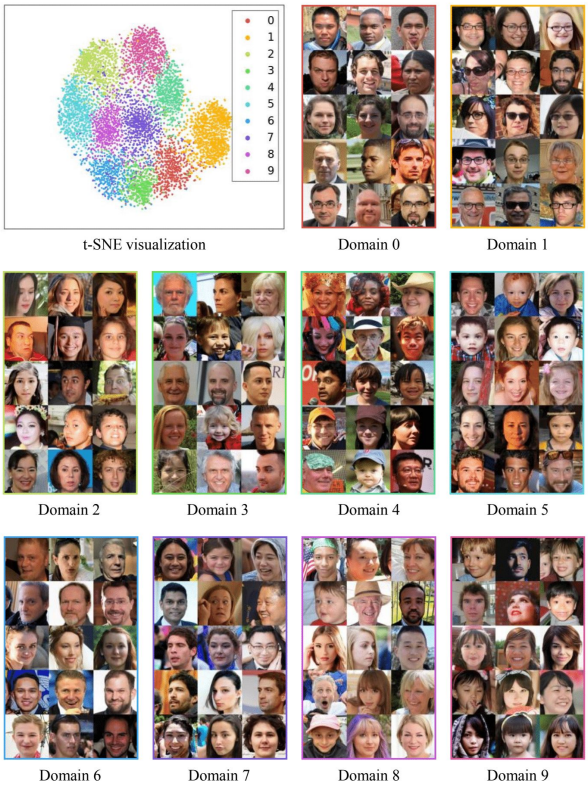


Figure 5: **t-SNE visualization and representative images of each domain for FFHQ.**

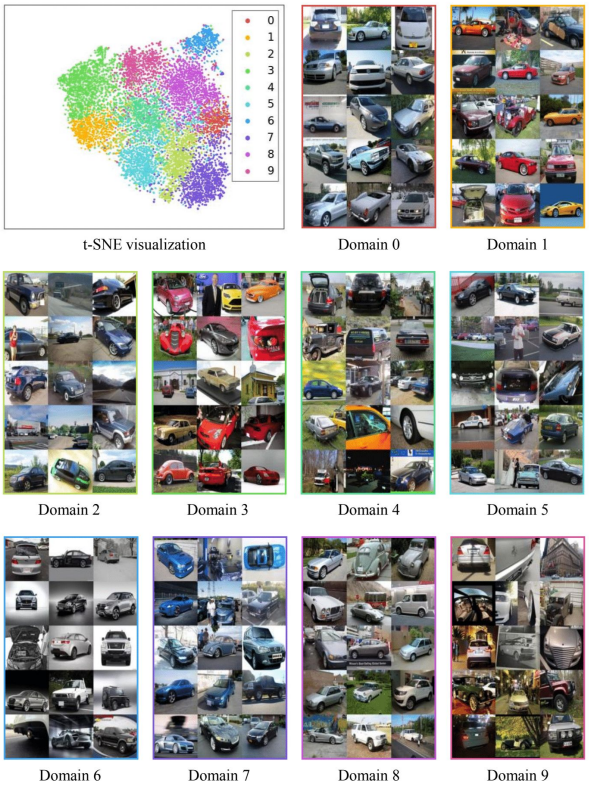


Figure 6: **t-SNE visualization and representative images of each domain for LSUN Car.**

10. Additional Comparison with FUNIT: AFHQ, LSUN Car and FFHQ

Figure 7: AFHQ Cat, unsupervised reference-guided image-to-image translation results of FUNIT and TUNIT. The content and the style are from the source and the reference, respectively. While FUNIT usually fails to reflect the style of the reference image, TUNIT generates the fake images with the style – color, fur texture.

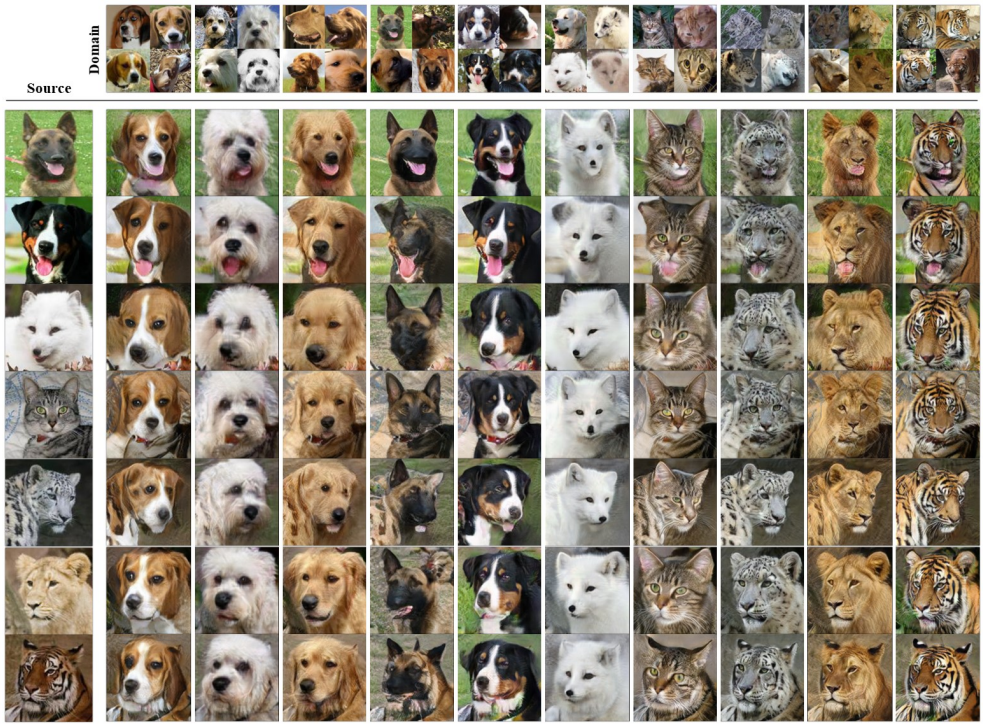
Figure 8: AFHQ Wild, unsupervised reference-guided image-to-image translation results of FUNIT and TUNIT. FUNIT rarely reflects the correct style of the reference image – the species, on the other hand, TUNIT translates the source image to the correct species.

Figure 9: LSUN Car, unsupervised reference-guided image-to-image translation results of FUNIT and TUNIT. While TUNIT generates plausible and changes the color of the source image to that of the reference image, FUNIT not also generates unrealistic image but also fails to changes the color.

Figure 10: FFHQ, unsupervised reference-guided image-to-image translation results of FUNIT and TUNIT. Our model, TUNIT can remove or add the glasses to the source while preserving the identity better than FUNIT. In addition, TUNIT can change the hair color (last column) and the hair style – especially, bang (fifth column). It is hard to specify the definition of domains in the results of FUNIT while domains of TUNIT are more interpretable.

11. Additional Results of TUNIT including semi-supervised setting

11.1. AnimalFaces-10



(a) Results guided by average style vectors

(b) Results guided by reference images

Figure 11: AnimalFaces-10, unsupervised image-to-image translation results.

11.2. AFHQ Cat

(a) Results guided by the average style code of each domain

(b) Results guided by reference images

Figure 12: AFHQ Cat, unsupervised image-to-image translation results.

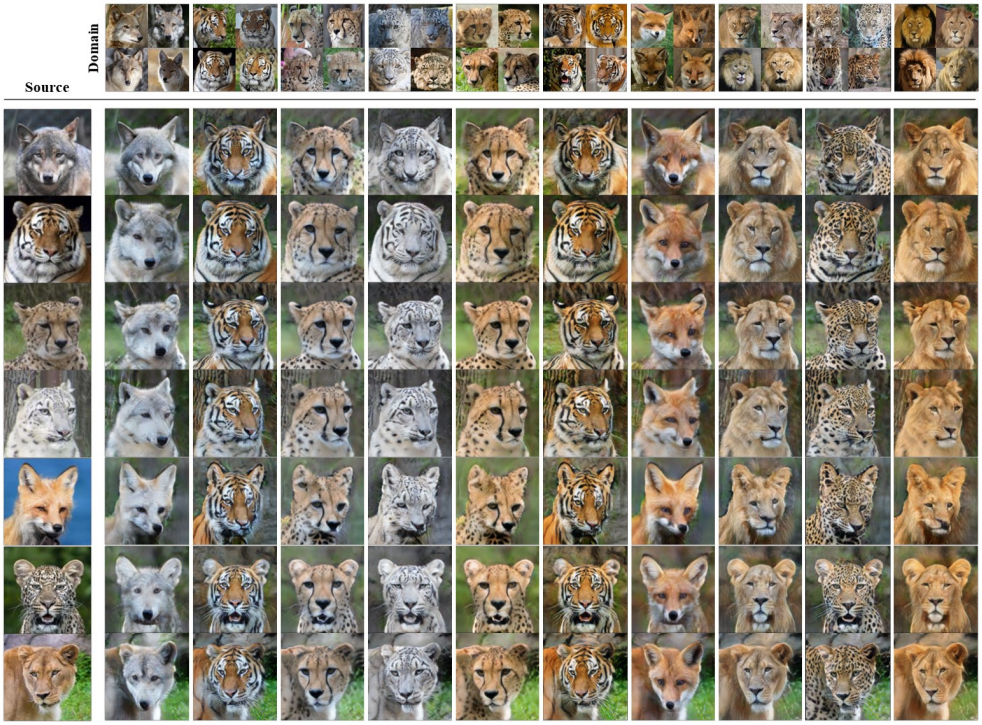
11.3. AFHQ Dog

(a) Results guided by the average style code of each domain

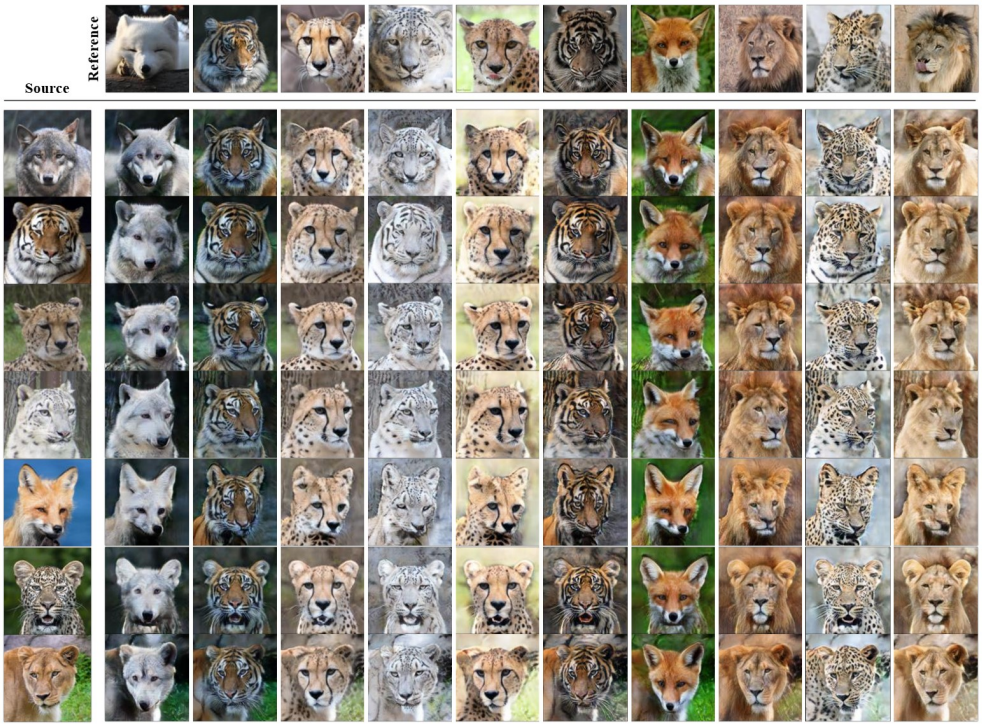
(b) Results guided by reference images

Figure 13: AFHQ Dogs, unsupervised image-to-image translation results.

11.4. AFHQ Wild



(a) Results guided by the average style code of each domain



(b) Results guided by reference images

Figure 14: AFHQ Wild, unsupervised image-to-image translation results.

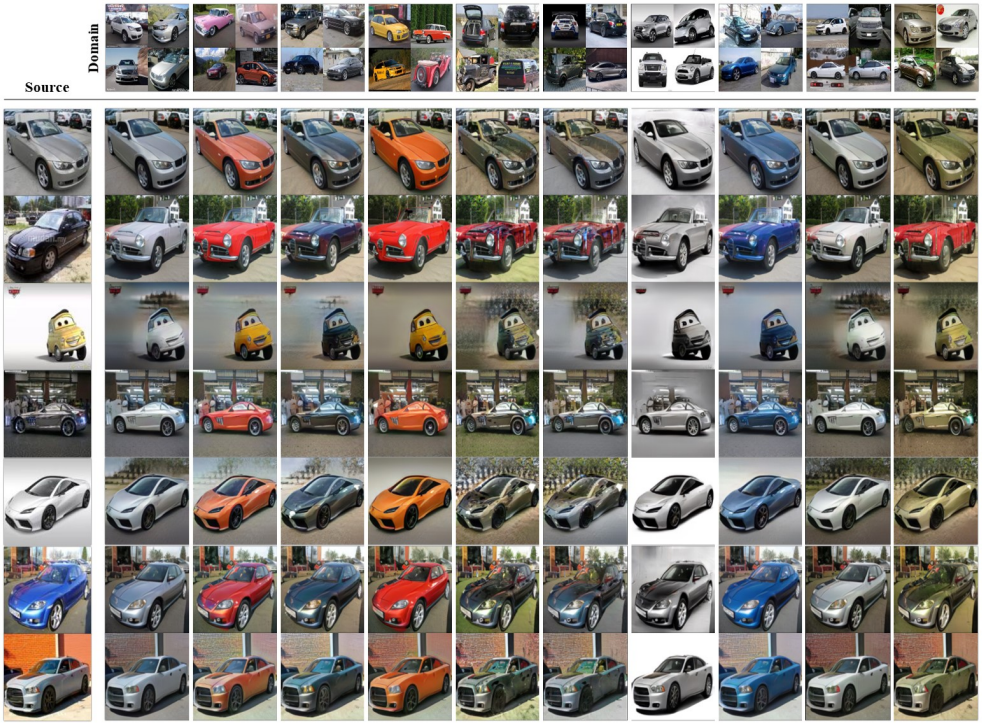
11.5. FFHQ

(a) Results guided by the average style code of each domain

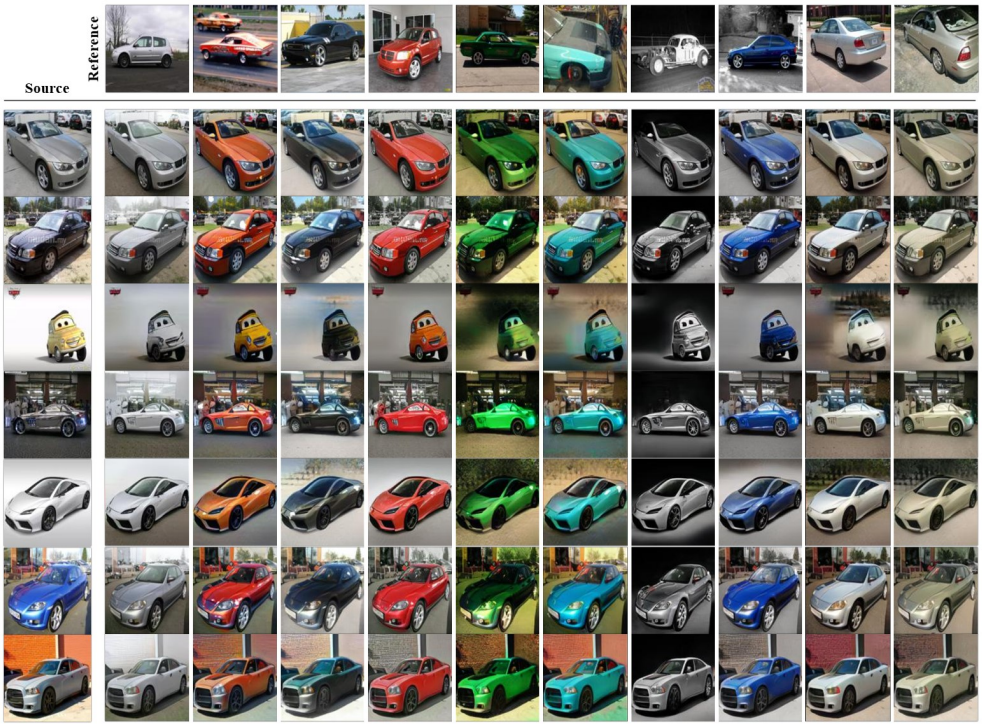
(b) Results guided by reference images

Figure 15: FFHQ, unsupervised image-to-image translation results.

11.6. LSUN Car



(a) Results guided by the average style code of each domain

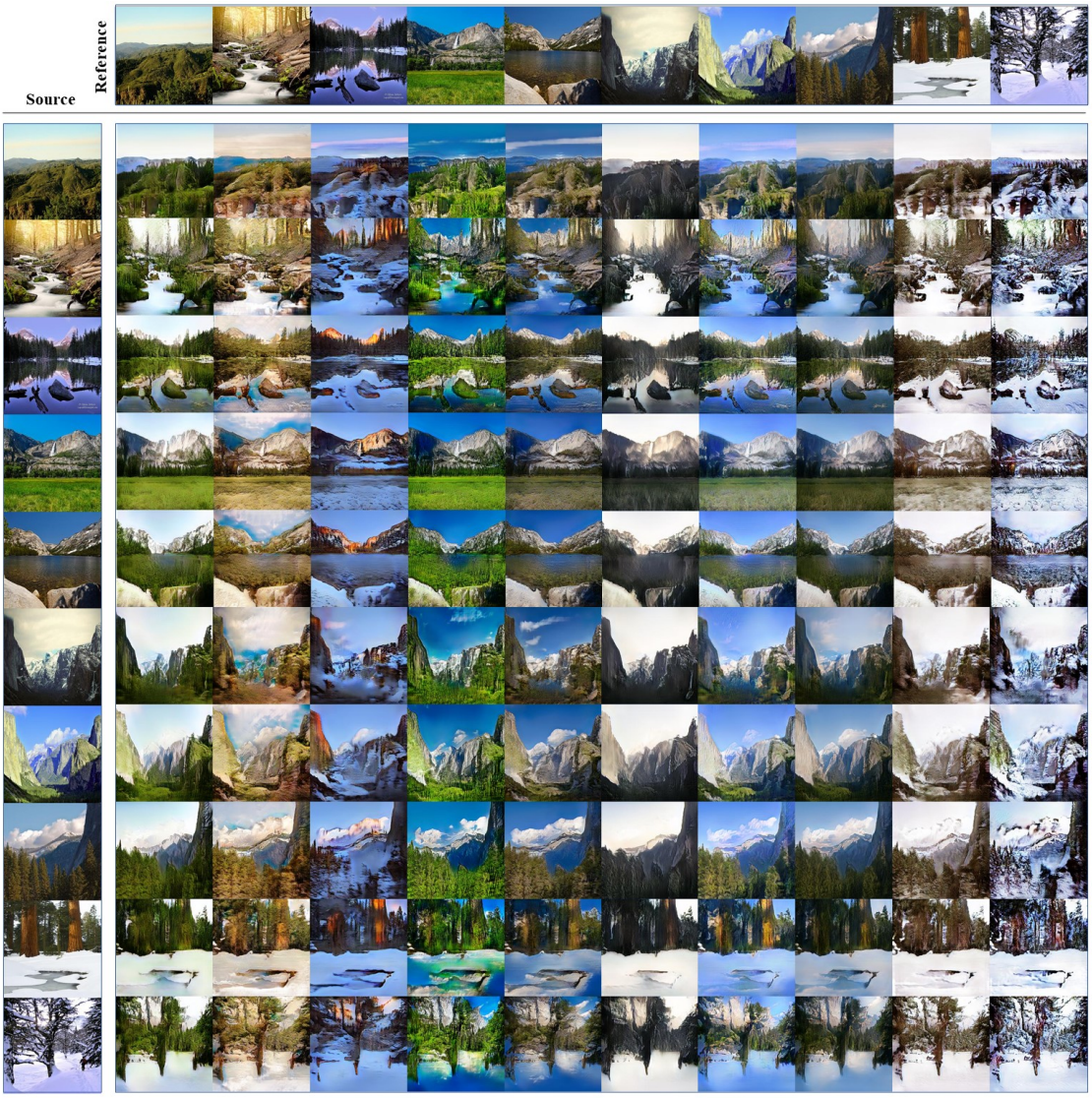


(b) Results guided by reference images

Figure 16: LSUN Car, unsupervised image-to-image translation results.

11.7. Summer2Winter (S2W)

(a) Results guided by the average style code of each domain



(b) Results guided by reference images

Figure 17: Summer2Winter (S2W), unsupervised image-to-image translation results.

11.8. Photo2Ukiyoe

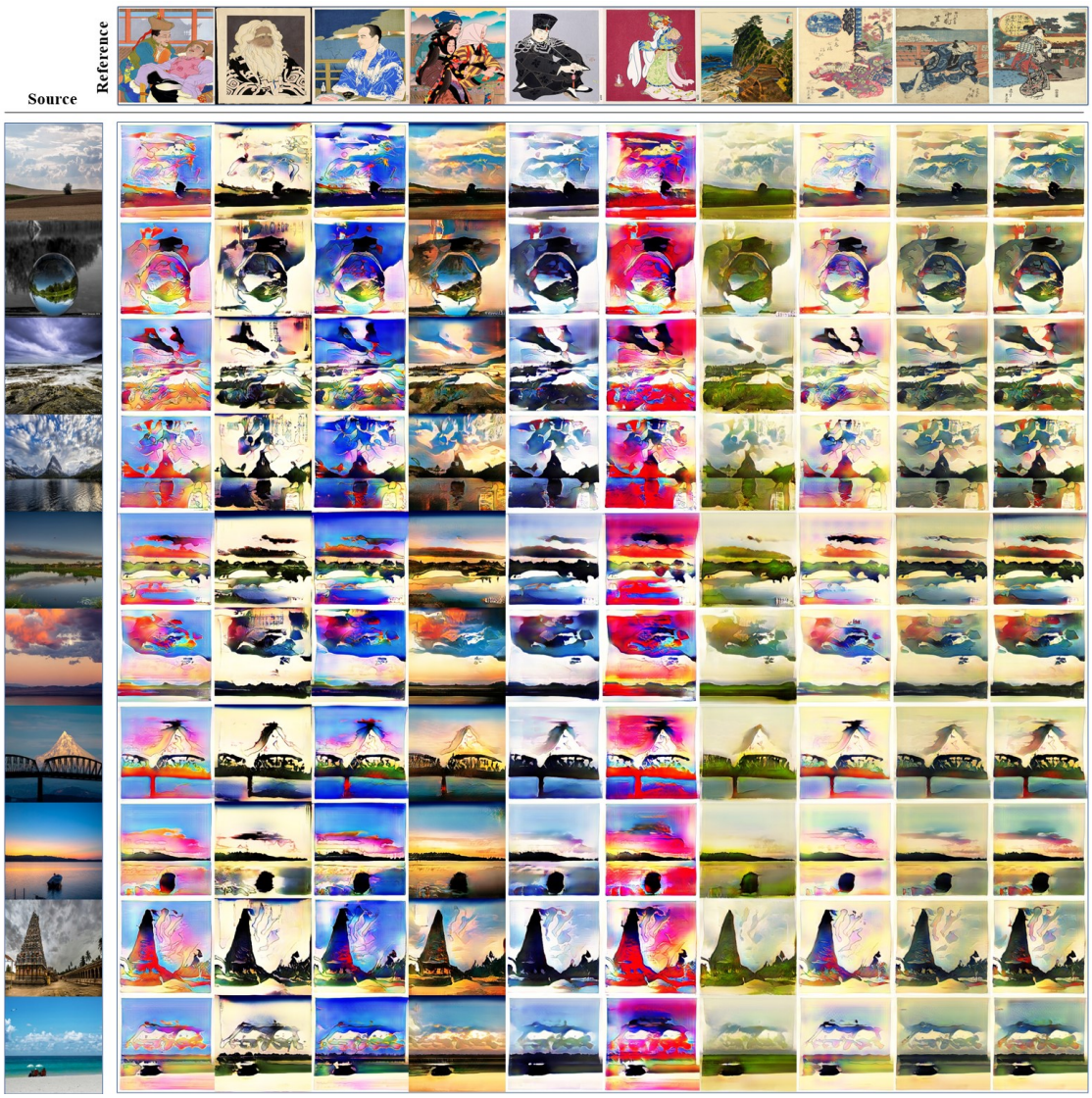


Figure 18: Photo2Ukiyoe, unsupervised image-to-image translation results.

11.9. AnimalFaces-149, comparing with SEMIT

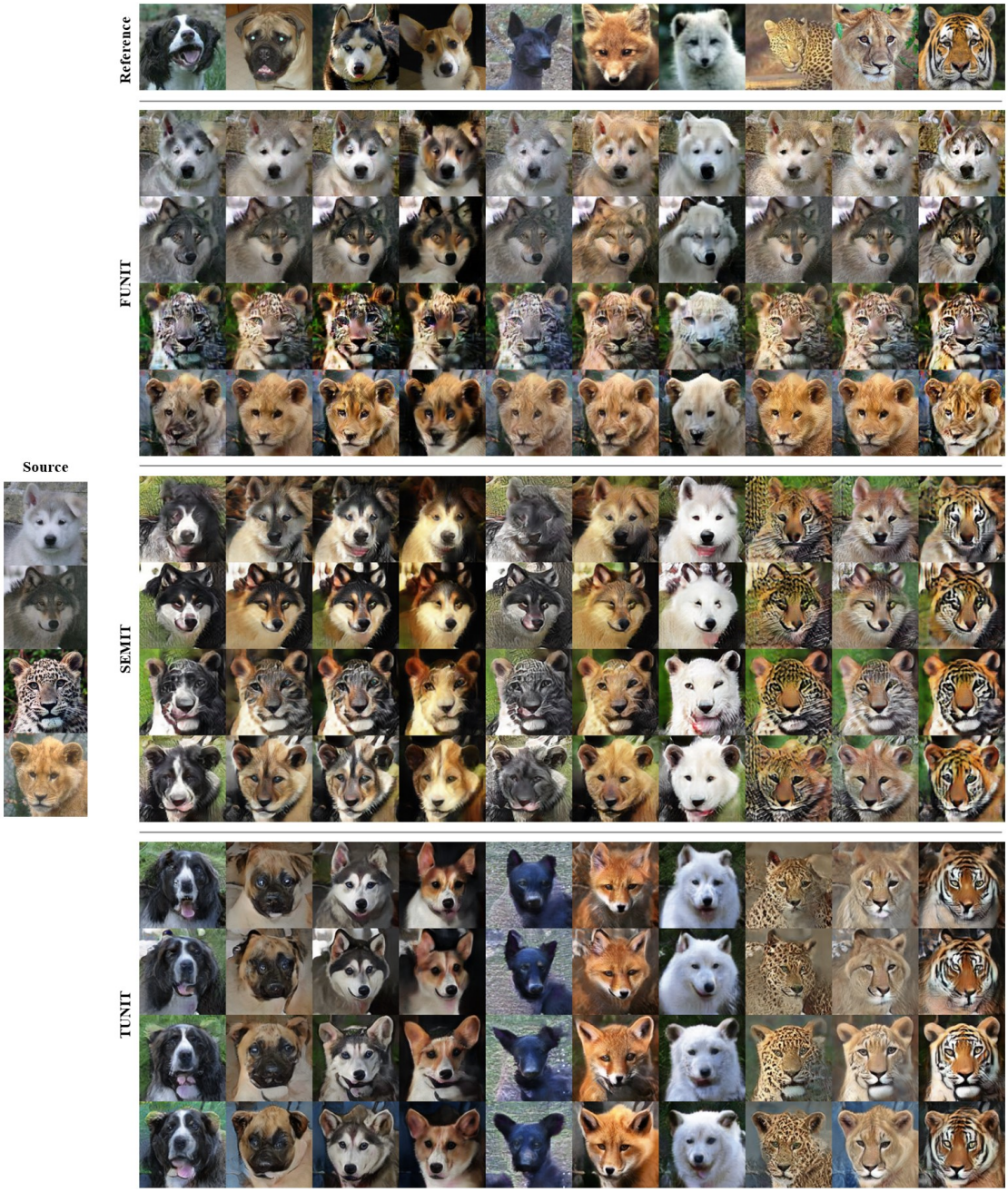


Figure 19: Qualitative comparison with SEMIT and FUNIT, semi-supervised translation results on AnimalFaces-149 (1% of labeled samples are used).

12. Difference between equation (2) and equation (4)

Equation (2) and (4) have similar forms – contrastive loss, but they are used for different purposes. We use equation (2) to improve the representation power of the guiding network, which affects the performance of the generator and the discriminator. On the other hand, equation (4) is used to enforce the generator to reflect the style of a reference image when translating a source image. To examine the effect of each loss, we train models without either equation (2) or (4) on AnimalFaces-10. The mFID score without equation (2) or (4) is 86.8 and 93.3, respectively. Both models are significantly worse than the original setting (47.7). It means that both equation (2) and (4) should be considered during training. In addition to the purpose, they are different in terms of the way to choose positive pairs. We use a real image and its randomly augmented version as a positive pair in equation (2) while we use the translated image and reference image as a positive pair. In summary, the role of equation (2) is to enhance the representation power of the guiding network and lead the guiding network to learn how to encode the style vector in terms of a style encoder while the role of equation (4) is to guide the generator to learn how to interpret the provided style vector as a form of the output image.

13. FID and LPIPS on unlabeled dataset

We also utilize LPIPS to evaluate the models in addition to FID and D&C. However, LPIPS is not proper to evaluate the loyalty for reflecting the reference image and the fidelity of images, we use LPIPS with FID. Figure 20 shows the result. It is clear that a model with high FID and LPIPS generates a noise-like image. Even if FID is low, a model with high LPIPS also fails to conduct the reference-guided image translation, because it does not preserve the structure of the source image. The model with low LPIPS and high FID might be an adversarial example of LPIPS. We generate the image via optimization on LPIPS. If a model exhibits low FID and LPIPS, it might not reflect the visual feature of the reference image enough. The simple combination of LPIPS and FID can detect several failed models but can not evaluate the loyalty for the reference image. We suggest that the rigorous way to combine several metrics for the quantitative evaluation of the reference-guided translation might be a interesting future work.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 1

[2] 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 equilib-

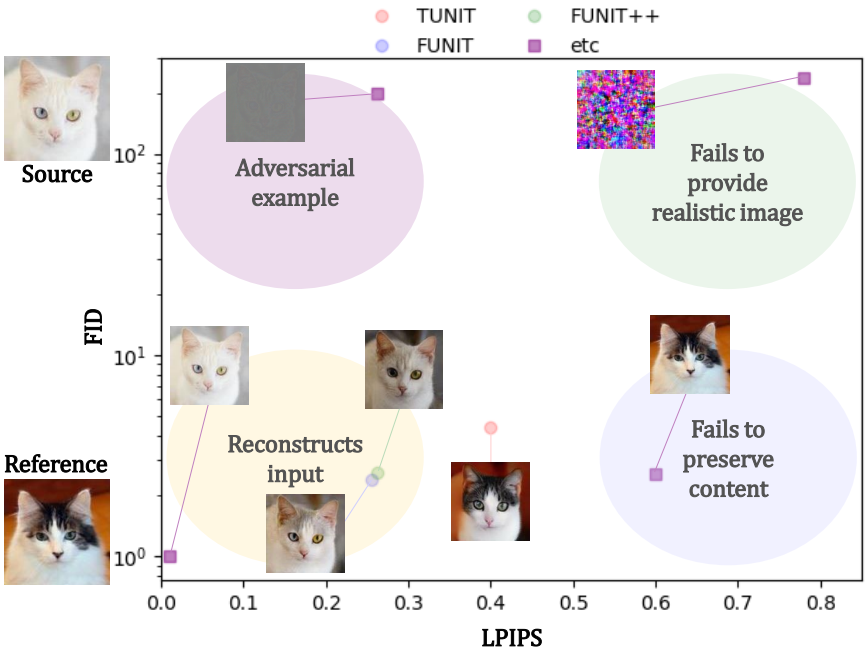


Figure 20: **LPIPS and FID of models and their status.**

rium. In *Advances in neural information processing systems*, pages 6626–6637, 2017. 1

[3] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. 2012. 1

[4] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 1

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[6] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017. 1

[7] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 2

[8] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for gans do actually converge? In *ICML*, 2018. 1

[9] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. 2020. 1

[10] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei A Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. *arXiv preprint arXiv:2007.00653*, 2020. 3

[11] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 1

[12] Saurabh Singh and Shankar Krishnan. Filter response normalization layer: Eliminating batch dependence in the training of deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11237–11246, 2020. 3

[13] Dustin Tran, Rajesh Ranganath, and David Blei. Hierarchical implicit models and likelihood-free variational inference. In *Advances in Neural Information Processing Systems*, pages 5523–5533, 2017. 1