COCO-FUNIT: Few-Shot Unsupervised Image Translation with a Content Conditioned Style Encoder

Kuniaki Saito[1,2], Kate Saenko[1], and Ming-Yu Liu[2]

Boston University[1] NVIDIA[2]
{keisaito,saenko}@bu.edu, mingyul@nvidia.com

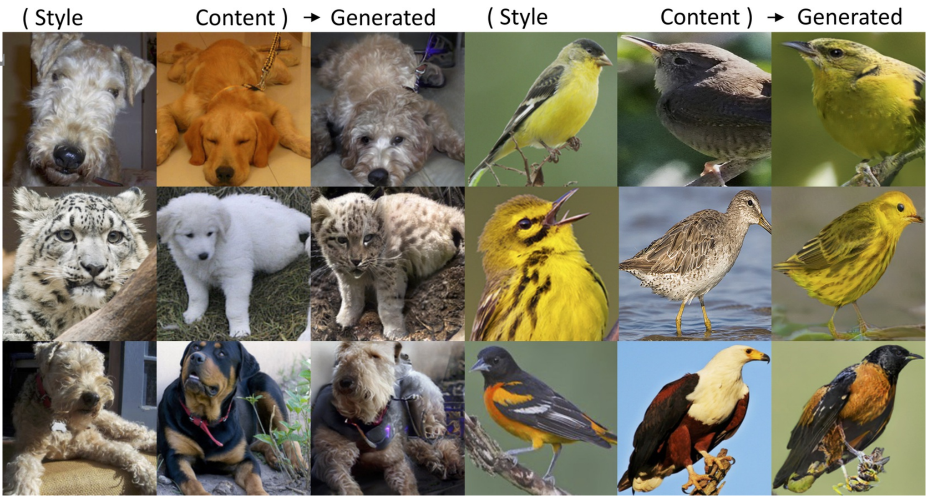

Fig. 1: Given as few as one style example image from an object class unseen during training, our model can generate a photorealistic translation of the input content image in the unseen domain.

Abstract. Unsupervised image-to-image translation intends to learn a mapping of an image in a given domain to an analogous image in a different domain, without explicit supervision of the mapping. Few-shot unsupervised image-to-image translation further attempts to generalize the model to an unseen domain by leveraging example images of the unseen domain provided at inference time. While remarkably successful, existing few-shot image-to-image translation models find it difficult to preserve the structure of the input image while emulating the appearance of the unseen domain, which we refer to as the *content loss* problem. This is particularly severe when the poses of the objects in the input and example images are very different. To address the issue, we propose a new few-shot image translation model, COCO-FUNIT, which computes the style embedding of the example images conditioned on the input image and a new module called the constant style bias. Through extensive experimental validations with comparison to the state-of-the-art, our model shows effectiveness in addressing the *content loss* problem. Code and pretrained models are available at `https://nvlabs.github.io/COCO-FUNIT/`.

Keywords: Image-to-image translation, Generative Adversarial Networks

1 Introduction

Image-to-Image translation [1,2] concerns learning a mapping that can translate an input image in one domain into an analogous image in a different domain. Unsupervised image-to-image translation [3,4,5,6,7,8,9] attempts to learn such a mapping without paired data. Thanks to the introduction of novel network architectures and learning objective terms, the state-of-the-art has advanced significantly in the past few years. However, while existing unsupervised image-to-image translation models can generate realistic translations, they still have several drawbacks. First, they require a large amount of images from the source and target domains for training. Second, they cannot be used to generate images in unseen domains. These limitations are addressed in the few-shot *unsupervised* image-to-image translation framework [10]. By leveraging example-guided episodic training, the few-shot image translation framework [10] learns to extract the domain-specific style information from a few example images in the unseen domain during test time, mixes it with the domain-invariant content information extracted from the input image, and generates a few-shot translation output as illustrated in Fig. 2.

However, despite showing encouraging results on relatively simple tasks such as animal face and flower translation, the few-shot translation framework [10] frequently generates unsatisfactory translation outputs when the model is applied to objects with diverse appearance, such as animals with very different body poses. Often, the translation output is not well-aligned with the input image. The domain invariant content that is supposed to remain unchanged disappears after translation, as shown in Fig. 3. We will call this issue the *content loss* problem. We hypothesize that solving the content loss problem would produce more faithful and photorealistic few-shot image translation results.

But why does the content loss problem occur? To learn the translation in an unsupervised manner, Liu *et al.* [10] rely on inductive bias injected by the network design and adversarial training [11] to transfer the appearance from the example images in the unseen domain to the input image. However, as there is no supervision, it is difficult to control what to be transferred precisely. Ideally, the transferred appearance should contain just the style. In reality, it often contains other information, such as the object pose.

In this paper, we propose a novel network architecture to counter the content loss problem. We design a style encoder called the *content-conditioned style encoder*, to hinder the transmission of task-irrelevant appearance information to the image translation process. In contrast to the existing style encoders, our

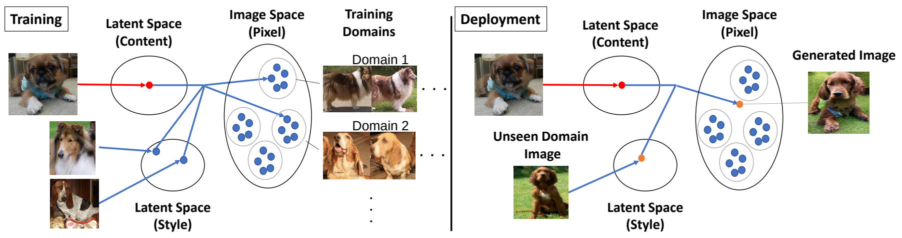

Fig. 2: Few-shot image-to-image translation. **Training.** The training set consists of many domains. We train a model to translate images between these domains. **Deployment.** We apply the trained model to perform few-shot image translation. Given a few examples from a test domain, we aim to translate a content image into an image analogous to the test class.

style code is computed by conditioning on the input content image. We use a new architecture design to limit the variance of the style code. We conduct an extensive experimental validation with a comparison to the state-of-the-art method using several newly collected and challenging few-shot image translation datasets. Experimental results, including both automatic performance metrics and user studies, verify the effectiveness of the proposed method in dealing with the content loss problem.

2 Related Works

Image-to-image translation. Most of the existing models are based on the Generative Adversarial Network (GAN) [11] framework. Unlike unconditional GANs [11,12,13,14,15], which learn to map random vectors to images, existing image-to-image translation models are mostly based on conditional GANs where they learn to generate a corresponding image in the target domain conditioned on the input image in the source domain. Depending on the availability of paired input and output images as supervision in the training dataset, image-to-image translation models can be divided into supervised [1,2,16,17,18,19,20,21,22,23,24,25] or unsupervised [3,4,5,6,7,8,9,26,27,28,29,30,31,32]. Our work falls in the category of unsupervised image-to-image translation. However, instead of learning a mapping between two specific domains, we aim at learning a flexible mapping that can be used to generate images in many unseen domains. Specifically, the mapping is only determined at test time, via example images. When using example images from a different unseen domain, the same model can generate images in the new unseen domain.

Multi-domain image translation. Several works [9,33,34,35] extend the unsupervised image translation to multiple domains. They learn a mapping between multiple seen domains, simultaneously. Our work differs from the multi-domain image translation works in that we aim to translate images to *unseen* domains.

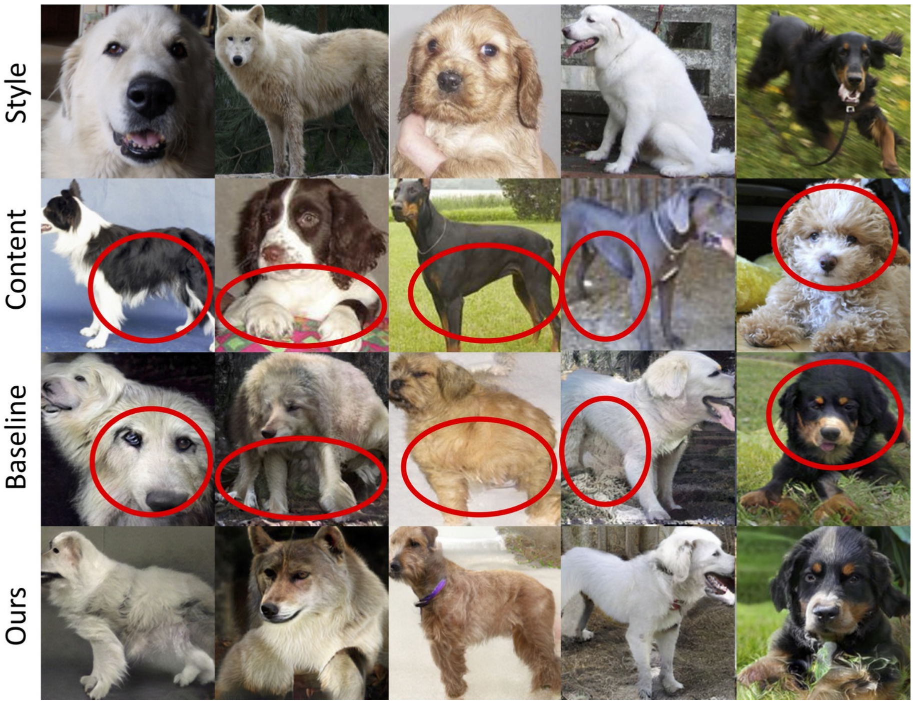

Fig. 3: Illustration of the *content loss* problem. The images generated by the baseline [10] fail to preserve domain invariant appearance information in the content image. The animals' bodies are sometimes merged with the background (column 3, & 4), scales of the generated body parts are sometimes inconsistent with the input (column 5), and some body parts absent in the content image show up (column 1 & 2). Our proposed method solves this "content loss" problem.

Few-shot image translation. Several few-shot methods are proposed to generate human images [36,25,37,38], scenes [25], or human faces [39,36,40] given a few instances and semantic layouts in a test time. These methods operate in the supervised setting. During training, they assume access to paired input (layout) and output data. Our work is most akin to the FUNIT work [10] as we aim to learn to generalize the translation to unseen domain without paired input and output data. We build on top of the FUNIT work where we first identify the *content loss* problem and then address it with a novel content-conditioned style encoder architecture.

Example-guided image translation refers to methods that generate a translation of an input conditioning on some example images. Existing works in this space [27,16,10] use a style encoder to extract style information from the example images. Our work is also an example-guided image translation method. However, unlike the prior works where the style code is computed independent of the input image, our style code is computed by conditioning on the input image, where we normalize the style code using the content to prevent over-transmission of the style information to the output.

Neural style transfer studies approaches to transfer textures from a painting to a real photo. While existing neural style transfer methods [41,42,43] can generalize to unseen textures, they cannot generalize to unseen shapes, necessary for image-to-image translation. Our work is inspired by these works, but we focus on generalizing the generation of both unseen shapes and textures, which is essential to few-shot unsupervised image-to-image translation.

3 Method

In this section, we start with a brief explanation of the problem setup, introduce the basic architecture, and then describe our proposed architecture. Throughout the paper, the two words, "class" and "domain", are used interchangeably since we treat each object class as a domain.

Problem setting. Fig. 2 provides an overview of the few-shot image translation problem [10]. Let X be a training set consists of images from K different domains. For each image in X, the class label is known. Note that we operate in the unsupervised setting where corresponding images between domains are *unavailable*. The few-shot image-to-image translation model learns to map a "content" image in one domain to an analogous image in the domain of the input "style" examples. In the test phase, the model sees a few example images from an unseen domain and performs the translation.

During training, a pair of content and style images x_c, x_k is randomly sampled. Let x_k denote a style image in domain k. The content image x_c can be from any domains in K. The generator G translates x_c into an image of class k ($\bar{\boldsymbol{x}}_k$) while preserving the content information of x_c.

$$\bar{\boldsymbol{x}}_k = G(x_c, x_k) \tag{1}$$

In the test phase, the generator takes style images from a domain unseen during training, which we call the target domain. The target domain can be any related domain, not included in K.

FUNIT baseline. FUNIT uses an example-guided conditional generator architecture as illustrated in the top-left of Fig. 4. It consists of three modules, 1) content encoder E_c, 2) style encoder E_s, and 3) image decoder F. E_c takes content image x_c as input and outputs content embedding z_c. E_s takes style image x_s as input and output style embedding z_s. Then, F generates an image using z_c and z_s, where z_s is used to generate the mean and scale parameters of adaptive instance normalization (AdaIN) layers [42] in F. The AdaIN design is based on the assumption that the domain-specific information can be governed by the first and second order statistics of the activation and has been used in several GAN frameworks [27,10,12]. We further note that when multiple example/style images are present. FUNIT extracts a style code from each image and

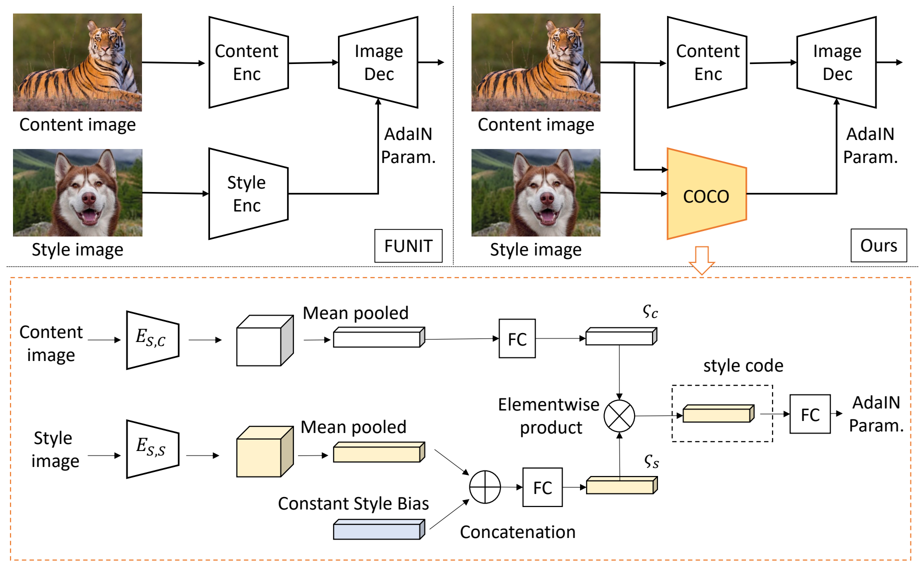

Fig. 4: **Top**. The FUNIT baseline [10] vs. our COCO-FUNIT. To highlight, we use a novel style encoder called the content-conditioned style encoder where the content image is also used in computing the style code for few-shot unsupervised image-to-image translation. **Bottom**. Detail design of the content-conditioned style encoder. Please refer to the main text for more details.

uses the average style code as the final input to F. To sum up, in FUNIT the image translation is formalized as follows,

$$z_c = E_c(x_c), \quad z_s = E_s(x_s), \quad \bar{\boldsymbol{x}} = F(z_c, z_s). \tag{2}$$

Content loss. As illustrated in Fig. 3, the FUNIT method suffers from the content loss problem—the translation result is not well-aligned with the input image. While a direct theoretical analysis is likely elusive, we conduct an empirical study, aiming at identify the cause of the content loss problem. As shown in Fig. 5, we compute different translation results of a content image based on a different style image where each of the style images is cropped from the same original style image. In the plot, we show variations of the deviation of the extracted style code due to different crops. Ideally, the plot should be constant as long as the crop covers sufficient appearance signature of the target class since that should be all required to generate a translation in the unseen domain. However, the FUNIT style encoder produces very different style codes as using different crops. Clearly, the style code contains other information about the style image such as the object pose. We hypothesize this is the cause of the content loss problem and revisit the translator network design for addressing it.

Content-conditioned style encoder (COCO). We hypothesize that the content loss problem can be mitigated if the style embedding is more robust to small variations in the style image. To this end, we design a new style encoder architecture, called the COntent-COnditioned style encoder (COCO). There are several distinctive features in COCO. The most obvious one is the conditioning in the content image as illustrated in the top-right of Fig. 4. Unlike the style encoder in FUNIT, COCO takes *both* content and style image as input. With this content-conditioning scheme, we create a *direct* feedback path during learning to let the content image influence how the style code is computed. It also helps reduce the direct influence of the style image to the extract style code.

The bottom part of Fig. 4 details the COCO architecture. First, the content image is fed into encoder $E_{S,C}$ to compute a spatial feature map. This content feature map is then mean-pooled and mapped to a vector ζ_c. Similarly, the style image is fed into encoder $E_{S,S}$ to compute a spatial feature map. The style feature map is then mean-pooled and concatenated with an input-independent bias vector, which we refer to as the constant style bias (CSB). Note that while the regular bias in deep networks is added to the activations, in CSB, the bias is concatenated with the activations. The CSB provides a fixed input to the style encoder, which helps compute a style code that is less sensitive to the variations in the style image. In the experiment section, we show that the CSB can also be used to control the type of appearance information that is transmitted from the style image. When the CSB is activated, mostly texture-based appearance information is transferred. Note that the dimension of the CSB is set to 1024 through the paper.

The concatenation of the style vector and the CSB is mapped to a vector ζ_s via a fully connected layer. We then perform an element-wise product operation to ζ_c and ζ_s, which is our final style code. The style code is then mapped to produce the AdaIN parameters for generating the translation. Through this element-wise product operation, the resulting style code is heavily influenced by the content image. One way to look at this mechanism is that it produces a customized style code for the input content image.

We use the COCO as a drop-in replacement for the style encoder in FUNIT. Let ϕ denote the COCO mapping. The translation output is then computed via

$$z_c = E_c(x_c),\ z_s = \phi(E_{s,s}(x_s), E_{s,c}(x_c)),\ \bar{\boldsymbol{x}} = F(z_c, z_s). \tag{3}$$

As shown in Fig. 5, the style code extracted by the COCO is more robust to variations in the style image. Note that we set $E_{S,C} \equiv E_C$ to keep the number of parameters in our model similar to that in FUNIT.

We note that the proposed COCO architecture shows only one way to generate the style code conditioned on the content and to utilize the CSD. Certainly, there exist other design choices that could potentially lead to better translation performance. However, since this is the first time these two components are used for the few-shot image-to-image translation task, we focus on analyzing their contribution in one specific design, i.e., our design. An exhaustive exploration is beyond the scope of the paper and is left for future work.

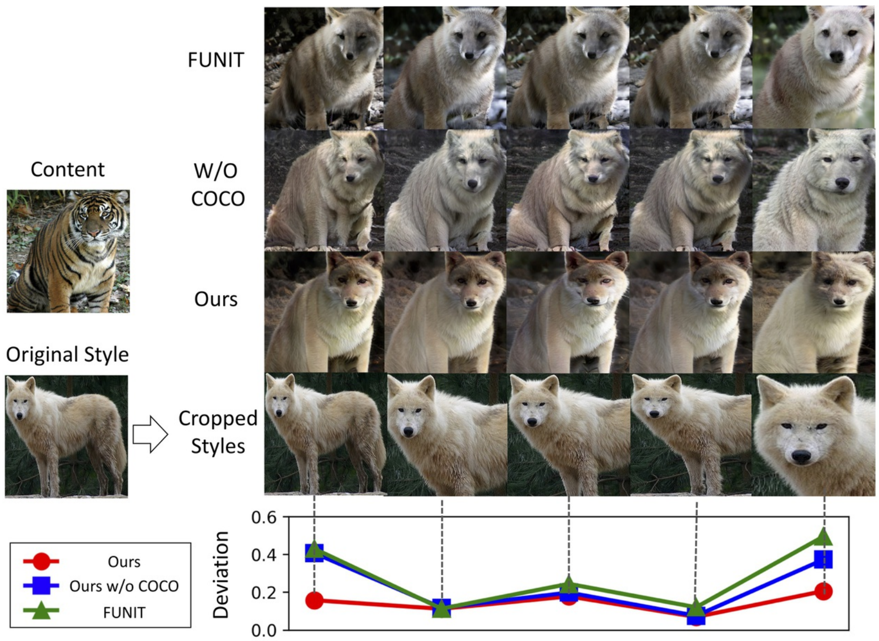

Fig. 5: We compare variations of the computed style codes due to variations in the style images for different methods. Note that for a fair comparison, in addition to the original FUNIT baseline [10], we create an improved FUNIT method by using our improved design for the content encoder, image decoder, and discriminator, which is termed "Ours w/o COCO". "Ours" is our full algorithm where we use COCO as a drop-in replacement for the style encoder in the FUNIT framework. In the bottom part of the figure, we plot the variations of the style code due to using different crops of a style image. Specifically, the style code for each style image is first extracted for each method. We then compute the mean of the style codes for each method. The magnitudes of the deviations from the mean style code are then plotted. Note that to calibrate the network weights in different methods, all the style codes are first normalized by the mean extracted from 500 style images for each method. As shown in the figure, "Ours" produces more consistent translation outputs, which is a direct consequence of a more consistent style code extraction mechanism.

In addition to the COCO, we also improve the design of the content encoder, image decoder, and discriminator in the FUNIT work [10]. For the content encoder and image decoder, we find that replacing the vanilla convolutional layers in the original design with residual blocks [44] improves the performance so does replacing the multi-task adversarial discriminator with the project-based discriminator [45]. In Appendix D, we report their individual contribution to the few-shot image translation performance.

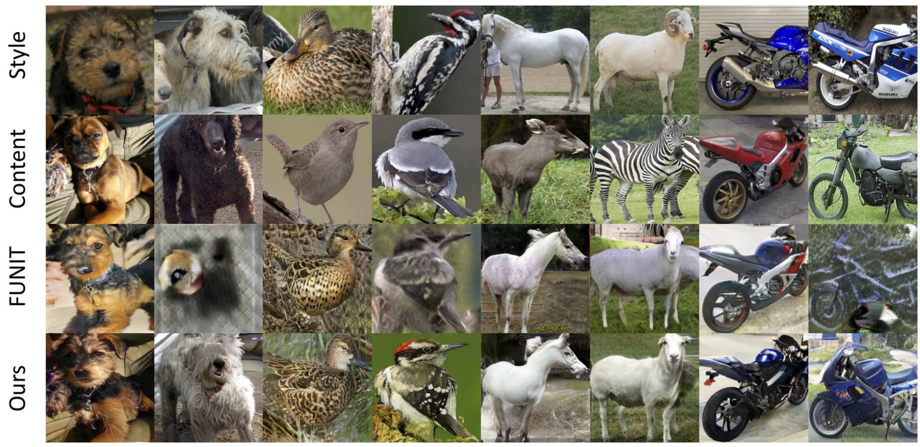

Fig. 6: Results on one-shot image-to-image translation. Column 1 & 2 are from the Carnivores dataset. Column 3 & 4 are from the Birds dataset. Column 5 & 6 are from the Mammals dataset. Column 7 & 8 are from the Motorbikes dataset.

Learning. We train our model using three objective terms. We use the GAN loss ($\mathcal{L}_{\mathrm{GAN}}(D,G)$) to ensure the realism of the generated images given the class of the style images. We use the image reconstruction loss ($\mathcal{L}_{\mathrm{R}}(G)$) to encourage the model to reconstruct images when both the content and the style are from the same domain. We use the discriminator feature matching loss ($\mathcal{L}_{\mathrm{FM}}(G)$) to minimize the feature distance between real and fake samples in the discriminator feature space, which has the effect of stabilizing the adversarial training and contributes to generating better translation outputs as shown in the FUNIT work. In Appendix B, we detail the computation of each loss. Overall the objective is

$$\min_{D}\max_{G} \mathcal{L}_{\mathrm{GAN}}(D,G) + \lambda_{\mathrm{R}}\mathcal{L}_{\mathrm{R}}(G) + \lambda_{\mathrm{F}}\mathcal{L}_{\mathrm{FM}}(G), \tag{4}$$

where λ_{R} and λ_{F} denote trade-off parameters for two losses. We set λ_{R} 0.1 and λ_{F} 1.0 in all of the experiments.

4 Experiments

We evaluate our method on several challenging datasets that contain large pose variations, part variations, and category variations. Unlike the FUNIT work, which focuses on translations between reasonably-aligned images or simple objects, our interest is in the translations between likely misaligned images of highly articulate objects. Throughout the experiments, we use 256×256 as our default image resolution for both inputs and outputs.

Implementation. We use Adam [46] with $lr = 0.0001$, $\beta_1 = 0.0$, and $\beta_2 = 0.999$ for all methods. Spectral normalization [47] is applied to the discriminator. The

Table 1: Results on the benchmark datasets.

Dataset	Method	mFID ↓	PAcc ↑	mIoU ↑	User Style Preference ↑	User Content Preference ↑
Carnivores	FUNIT	147.8	59.8	44.6	16.5	11.9
	Ours	**107.8**	**66.5**	**52.1**	**83.5**	**88.1**
Mammals	FUNIT	245.8	35.3	23.3	23.6	27.8
	Ours	**109.3**	**48.8**	**35.5**	**76.4**	**72.2**
Birds	FUNIT	89.2	52.4	37.2	38.5	37.5
	Ours	**74.6**	**53.3**	**38.3**	**61.5**	**62.5**
Motorbikes	FUNIT	275.0	85.6	73.8	17.8	17,4
	Ours	**56.2**	**94.6**	**90.3**	**82.2**	**82.6**

Table 2: Ablation study on the Carnivores and Birds dataset. "Ours w/o CC" represents a baseline where the content conditioning part in COCO is removed. "Ours w/o CSB" represents a baseline where the CSB is removed. Detailed architecture of these baselines are given in Appendix A

Method	Carnivores			Birds		
	mFID↓	PAcc↑	mIou ↑	mFID↓	PAcc↑	mIou ↑
Ours w/o COCO	**99.6**	62.5	47.8	**68.8**	52.8	37.9
Ours w/o CSB	107.1	61.8	46.9	74.1	52.5	37.7
Ours w/o CC	110.0	**66.7**	**52.1**	75.3	52.8	37.9
Ours	107.8	66.5	**52.1**	74.6	**53.3**	**38.3**

final generator is a historical average version of the intermediate generators [13] where the update weight is 0.001. We train the model for 150,000 iterations in total. For every competing model, we compute the scores every 10,000 iterations and report the scores of the iteration that achieves the smallest mFID. Each training batch consists of 64 content images, which are evenly distributed on a DGX machine with 8 V100 GPUs, each with 32GB RAM.

Datasets. We benchmark our method using 4 datasets. Each of the dataset contains objects with diverse poses, parts, and appearances.

- *Carnivores.* We build the dataset using images from the ImageNet dataset[48]. We pick up images from the 149 carnivorous animals and used 119 as the source/seen classes and 30 as the target/unseen classes.
- *Mammals.* We collect 152 classes of herbivore animal images using Google image search and combine them with the Carnivores dataset to build the Mammals dataset. Out of the 301 classes, 236 classes are used for the source/seen and the rest is used for the target/unseen.
- *Birds.* We collect 205 classes of bird images using Google image search. 172 classes are used for training and the rest is used for the testing.

- *Motorbikes.* We also collected 109 classes of motorbike images in the same way. 92 classes are used as the source and the rest is used for the target.

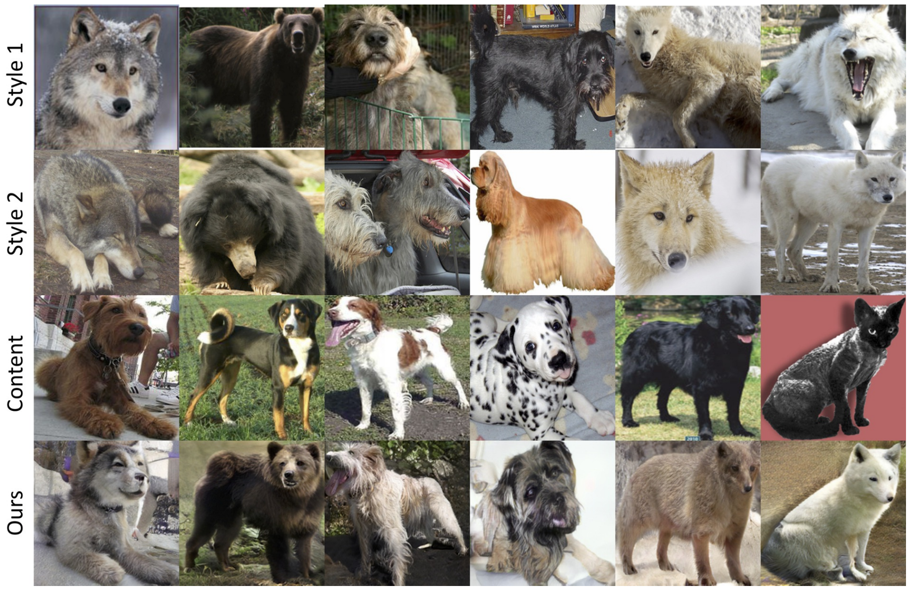

Fig. 7: Two-shot image translation results on the Carnivores dataset.

Evaluation protocol. For each dataset, we train a model using the source classes mentioned above and test the performance on the target classes for each competing methods. In the test phase, we randomly sample 25,000 content images and pair each of them with a few style images from a target class to compute the translation. Unless specified otherwise, we use the one-shot setting for performance evaluation as it is the most challenging few-shot setting. We evaluate the quality of the translated images using various metrics as explained below.

Performance metrics. Ideally, a translated image should keep the structure of the input content image, such as the pose or scale of body parts, unchanged when emulating the appearances of the unseen domain. Existing work mainly focused on the style transfer evaluation because the experiments are performed on well-aligned images or images of simple objects. To consider both the style translation and content preservation, we employ the following metrics. First, we evaluate the style transfer by measuring distance between the distribution of the translated images and the distribution of the real images in the unseen domain using mFID. Second, the content preservation is evaluated by measuring correspondence between a content and a translated image by matching their segmentation masks using mIou and PAcc. Third, we conduct a user study to compute human preference scores on both the style transfer and content preser-

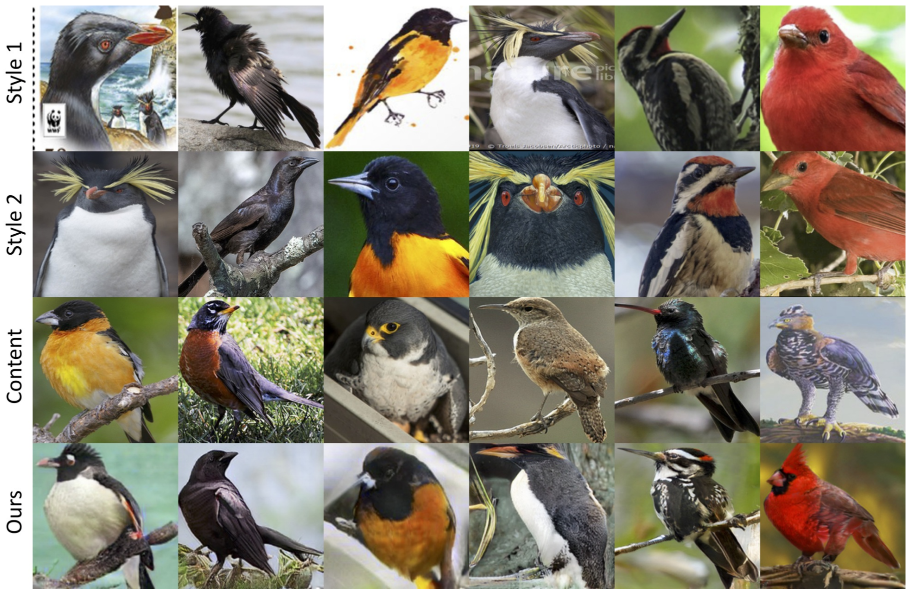

Fig. 8: Two-shot image translation results on the Birds dataset.

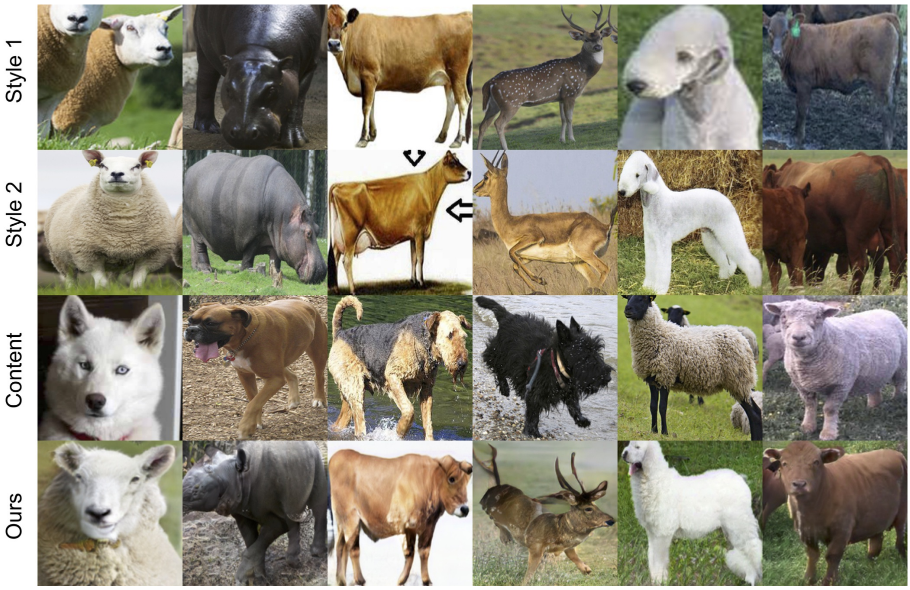

Fig. 9: Two-shot image translation results on the Mammals dataset.

vation of the translation results. The details of the performance metrics are given in Appendix C.

Baseline. We compare our method with the FUNIT method because it outperforms many baselines with a large margin as described in Liu *et al.* [10]. Therefore, a direct comparison with this baseline can verify the effectiveness of the proposed method for the few-shot image-to-image translation task.

Main results. The comparison results is summarized in Table 1. As shown, our method outperforms FUNIT by a large margin in all the datasets on both automatic metrics and human preference scores. This validates the effectiveness of our method for few-shot unsupervised image-to-image translation. Fig. 3 and 6 compare the one-shot translation results computed by the FUNIT method and our approach. We find images generated by the FUNIT method contain many artifacts while our method can generate photorealistic and faithful translation outputs. In Fig. 7, 8, and 9, we further visualize two-shot translation results. More visualization results are provided in Appendix D.

Ablation study. In Table 2, we ablate modules in our architecture and measure their impact on the few-shot translation performance using the Carnivores and Birds datasets. Now, let us walk through the results. First, we find using the CSB improve content preservation scores (”Ours w/o CSB” vs ”Ours”), reflected by the better PAcc and mIoU scores achieved. Second, using content conditioning improves style transferring (”Ours w/o CC” vs ”Ours”), reflected by the better mFID scores achieved. We also note that despite ”Ours w/o COCO” achieves a better mFID, it is in the expense of large content loss. We note that the numbers of parameters of the translation model are 241M for Ours w/o COCO and 242M for Ours. Therefore, adding the COCO module only increases the size of the network by a tiny amount.

In Appendix D, we present additional ablation study on the network architecture design choices. We also analyze impacts of the translation performance with respect to the number of example images available at test time and the number of style classes available at the training time.

Effect of the CSB. We conduct an experiment to understand how the CSB designed added to our COCO influences the translation results. Specifically, during testing, we multiply the CSB with a scalar λ. We then change the λ value to visualize its effect as shown in Fig. 10. Interestingly, different values of λ generate different translation results. When the value is small, the model mostly changes the texture of the content image. With a large λ value, both the shape and texture are changed.

Unseen style blending. Here, we show an application where we combine two style images from two unseen domains to create a new unseen domain. Specifically, we first extract two style codes from two images from two different unseen domains. We then mix their styles by linear interpolating the style codes. The results are shown in Fig. 11 where the leftmost image is the content and row indicated by `S1` and `S2` are the two style images. We find the intermediate style codes render plausible translation results.

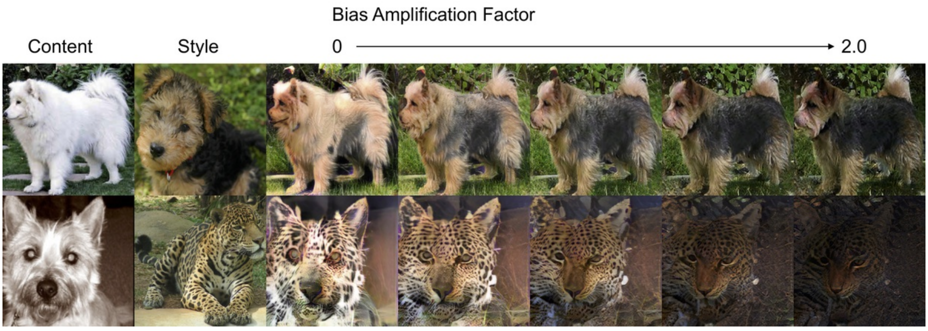

Fig. 10: By changing the amplification factor λ of the CSB, our model generates different translation outputs for the same pair of content and style images.

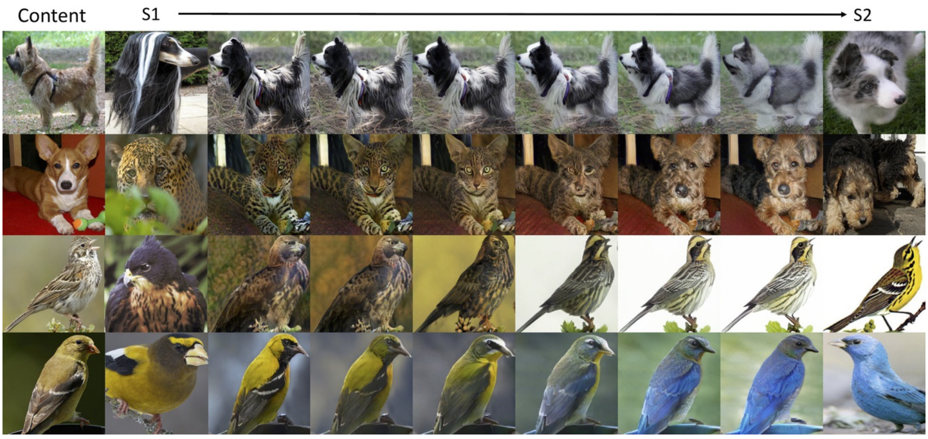

Fig. 11: We interpolate the style codes from two example images from two different unseen domains. Our model can generate photorealistic results using these interpolated style codes. More results are in the supplementary materials.

Failure cases. While our approach effectively addresses the content loss problem, it still have several failure modes. We discuss these failure modes in Appendix D.

5 Conclusion

We introduced the COCO-FUNIT architecture, a new style encoder for few-shot image-to-image translation that extracts the style code from the example images from the unseen domain conditioning on the input content image and uses a constant style bias design. We showed that the COCO-FUNIT can effectively address the content loss problem, proven challenging for few-shot image-to-image-translation.

Acknowledgements. We would like to thank Jan Kautz for his insightful feedback on our project. Kuniaki Saito and Kate Saenko were supported by Honda, DARPA and NSF Award No. 1535797. Kuniaki Saito contributed to this work during his internship at NVIDIA.

References

1. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017)
2. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional GANs. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)
3. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: IEEE International Conference on Computer Vision (ICCV). (2017)
4. Liu, M.Y., Breuel, T., Kautz, J.: Unsupervised image-to-image translation networks. In: Advances in Neural Information Processing Systems (NeurIPS). (2017)
5. Liang, X., Lee, L., Dai, W., Xing, E.P.: Dual motion GAN for future-flow embedded video prediction. In: Advances in Neural Information Processing Systems (NeurIPS). (2017)
6. Kim, T., Cha, M., Kim, H., Lee, J.K., Kim, J.: Learning to discover cross-domain relations with generative adversarial networks. In: International Conference on Machine Learning (ICML). (2017)
7. Liu, M.Y., Tuzel, O.: Coupled generative adversarial networks. In: Advances in Neural Information Processing Systems (NeurIPS). (2016)
8. Taigman, Y., Polyak, A., Wolf, L.: Unsupervised cross-domain image generation. In: International Conference on Learning Representations (ICLR). (2017)
9. Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)
10. Liu, M.Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., Kautz, J.: Few-shot unsupervised image-to-image translation. In: IEEE International Conference on Computer Vision (ICCV). (2019)
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. In: Advances in Neural Information Processing Systems (NeurIPS). (2014)
12. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
13. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: International Conference on Learning Representations (ICLR). (2018)
14. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein GANs. In: Advances in Neural Information Processing Systems (NeurIPS). (2017)
15. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Smolley, S.P.: Least squares generative adversarial networks. In: IEEE International Conference on Computer Vision (ICCV). (2017)

16. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
17. Liu, X., Yin, G., Shao, J., Wang, X., et al.: Learning to predict layout-to-image conditional convolutions for semantic image synthesis. In: Advances in Neural Information Processing Systems (NeurIPS). (2019)
18. Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In: IEEE International Conference on Computer Vision (ICCV). (2017)
19. Qi, X., Chen, Q., Jia, J., Koltun, V.: Semi-parametric image synthesis. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)
20. Zhu, J.Y., Zhang, R., Pathak, D., Darrell, T., Efros, A.A., Wang, O., Shechtman, E.: Toward multimodal image-to-image translation. In: Advances in Neural Information Processing Systems (NeurIPS). (2017)
21. Zhu, S., Urtasun, R., Fidler, S., Lin, D., Change Loy, C.: Be your own prada: Fashion synthesis with structural coherence. In: IEEE International Conference on Computer Vision (ICCV). (2017)
22. Zhao, B., Meng, L., Yin, W., Sigal, L.: Image generation from layout. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
23. Wang, C., Zheng, H., Yu, Z., Zheng, Z., Gu, Z., Zheng, B.: Discriminative region proposal adversarial networks for high-quality image-to-image translation. In: European Conference on Computer Vision (ECCV). (2018)
24. Wang, T.C., Liu, M.Y., Zhu, J.Y., Liu, G., Tao, A., Kautz, J., Catanzaro, B.: Video-to-video synthesis. In: Advances in Neural Information Processing Systems (NeurIPS). (2018)
25. Wang, M., Yang, G.Y., Li, R., Liang, R.Z., Zhang, S.H., Hall, P.M., Hu, S.M.: Example-guided style-consistent image synthesis from semantic labeling. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
26. Pumarola, A., Agudo, A., Martinez, A.M., Sanfeliu, A., Moreno-Noguer, F.: Ganimation: Anatomically-aware facial animation from a single image. In: European Conference on Computer Vision (ECCV). (2018)
27. Huang, X., Liu, M.Y., Belongie, S., Kautz, J.: Multimodal unsupervised image-to-image translation. In: European Conference on Computer Vision (ECCV). (2018)
28. Lee, H.Y., Tseng, H.Y., Huang, J.B., Singh, M., Yang, M.H.: Diverse image-to-image translation via disentangled representations. In: European Conference on Computer Vision (ECCV). (2018)
29. Shen, Z., Huang, M., Shi, J., Xue, X., Huang, T.S.: Towards instance-level image-to-image translation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
30. Gokaslan, A., Ramanujan, V., Ritchie, D., In Kim, K., Tompkin, J.: Improving shape deformation in unsupervised image-to-image translation. In: European Conference on Computer Vision (ECCV). (2018)
31. Benaim, S., Wolf, L.: One-shot unsupervised cross domain translation. In: Advances in Neural Information Processing Systems (NeurIPS). (2018)
32. AlBahar, B., Huang, J.B.: Guided image-to-image translation with bi-directional feature transformation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
33. Anoosheh, A., Agustsson, E., Timofte, R., Van Gool, L.: Combogan: Unrestrained scalability for image domain translation. arXiv preprint arXiv:1712.06909 (2017)
34. Hui, L., Li, X., Chen, J., He, H., Yang, J.: Unsupervised multi-domain image translation with domain-specific encoders/decoders. arXiv preprint arXiv:1712.02050 (2017)

35. Choi, Y., Uh, Y., Yoo, J., Ha, J.W.: Stargan v2: Diverse image synthesis for multiple domains. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2020)
36. Wang, T.C., Liu, M.Y., Tao, A., Liu, G., Kautz, J., Catanzaro, B.: Few-shot video-to-video synthesis. In: Advances in Neural Information Processing Systems (NeurIPS). (2019)
37. Siarohin, A., Lathuilire, S., Tulyakov, S., Ricci, E., Sebe, N.: Animating arbitrary objects via deep motion transfer. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
38. Han, X., Wu, Z., Wu, Z., Yu, R., Davis, L.S.: Viton: An image-based virtual try-on network. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)
39. Zakharov, E., Shysheya, A., Burkov, E., Lempitsky, V.: Few-shot adversarial learning of realistic neural talking head models. In: IEEE International Conference on Computer Vision (ICCV). (2019)
40. Gu, Q., Wang, G., Chiu, M.T., Tai, Y.W., Tang, C.K.: Ladn: Local adversarial disentangling network for facial makeup and de-makeup. In: IEEE International Conference on Computer Vision (ICCV). (2019)
41. Gatys, L.A., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. In: Advances in Neural Information Processing Systems (NeurIPS). (2015)
42. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: IEEE International Conference on Computer Vision (ICCV). (2017)
43. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Universal style transfer via feature transforms. In: Advances in Neural Information Processing Systems (NeurIPS). (2017)
44. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016)
45. Miyato, T., Koyama, M.: cGANs with projection discriminator. In: International Conference on Learning Representations (ICLR). (2018)
46. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR). (2015)
47. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations (ICLR). (2018)
48. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2009)
49. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
50. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: International Conference on Machine Learning (ICML). (2018)
51. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In: Advances in Neural Information Processing Systems (NeurIPS). (2017)
52. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016)
53. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)

54. Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision (IJCV) (2015)

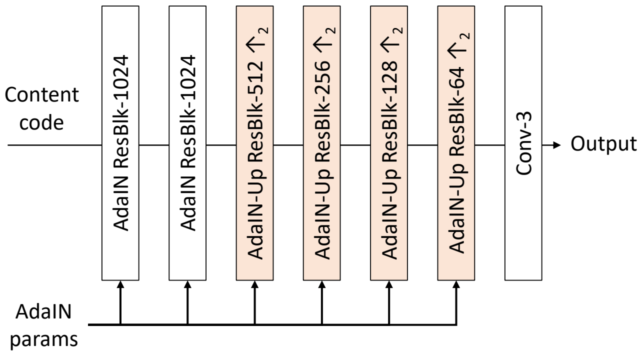

Fig. 14: Architecture of the image decoder F.

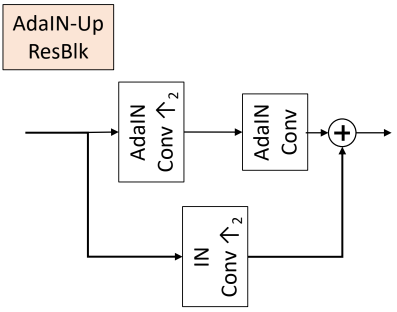

Fig. 15: Architecture of the AdaIN-Up residual block.

A Network Architecture

Here, we present the design of our various sub-networks described in the main paper. Note that we apply the ReLU nonlinearity to all the convolutional layers in the sub-networks in the generator.

Content Encoder. For the content encoder E_c, which is used to obtain a content code to be fed into the decoder for image translation, we mostly follow the content encoder architecture in FUNIT [10]. The only modification we apply is to add one more downsampling layer since the image resolution used in the experiments in the FUNIT paper is 128×128 and the image resolution in our experiments is 256×256. The detailed design is given in Fig. 12.

Content-Conditioned Style Encoder (COCO). In COCO, we have two sub-networks, $E_{s,c}$ and $E_{s,s}$ as shown in Fig. 4 of the main paper. Similar to E_c, $E_{s,c}$ is used to extract content information from the input image. The difference is that the content code extracted by $E_{s,c}$ is used to compute the style code. In our design, to reduce the number of parameters, we let $E_{s,c}$ be identical to E_c. For $E_{s,s}$, we follow the design in FUNIT [10] as shown in Fig. 13.

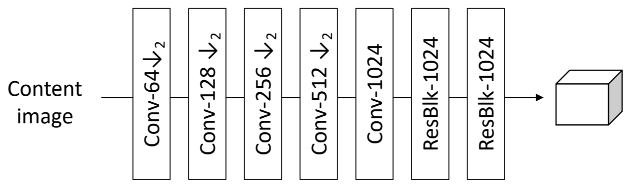

Fig. 12: Architecture of the content encoder E_c.

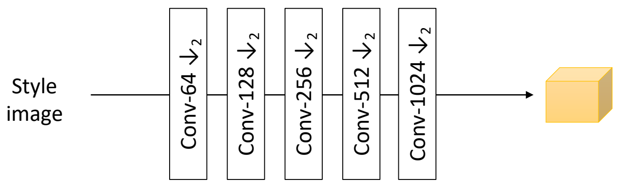

Fig. 13: Architecture of the sub-network $E_{s,s}$ in the content-conditioned style encoder (COCO).

Image decoder. We propose several modifications to the image generator in FUNIT. Specifically, we replace several convolutional blocks using a new kind of residual block, which we will call the AdaIN-Up Residual Block. We visualize the image decoder in Fig. 14 where the modification is highlighted in the light orange color. The detail of the AdaIN-Up Residual Block is visualized in Fig. 15. The residual block consists of a skip connection with upsampling convolution layer followed by an instance normalization layer [49] and a upsampling convolutions with AdaIN [42].

Discriminator. Our discriminator is a patch-based projection discriminator [45]. It utilizes the Leaky ReLU nonlinearity. The spectral normalization [47] is utilized in every layer. The discriminator consists of one convolutional layer followed by 10 activation first residual blocks [44]. The last layer is modified for class conditional projection projection. The architecture is illustrated via the following chain of operations:
`Conv-64` → `ResBlk-128` → `ResBlk-128` → `AvePool2x2` → `ResBlk-256` → `ResBlk-256` → `AvePool2x2` → `ResBlk-512` → `ResBlk-512` → `AvePool2x2`→ `ResBlk-1024` → `ResBlk-1024` → `AvePool2x2` → `ResBlk-1024` → `ResBlk-1024` → (`Conv-1`, `Conv-`$||\mathbb{S}||$) where $||\mathbb{S}||$ is the number of source classes. The last (`Conv-1`, `Conv-`$||\mathbb{S}||$) denotes the project operation of the patch-based hidden representation and the class embedding.

B Learning Objective Function

We train our model using a similar objective function as in the FUNIT work. Below, we first describe the individual objective terms and then present the overall optimization problem. Note that we do not utilize the gradient penalty term [50] used in the FUNIT work.

Adversarial loss. $\mathcal{L}_{\text{GAN}}(D, G)$ denotes class conditional GAN loss. We use the loss to ensure both photorealism and domain-faithfulness of image translation. Following the projection discriminator design [45], we use the hinge version of the adversarial loss.

Reconstruction loss. The loss encourages the model to reconstruct images when both the content and the style are from the same domain. This loss helps regularize the learning and is given by

$$\mathcal{L}_{\text{R}}(G) = E_{x_c}[||x_c - \bar{\boldsymbol{x_c}}||_1], \tag{5}$$

where $\bar{\boldsymbol{x_c}} = G(x_c, x_c)$.

Discriminator Feature matching loss. Minimizing the feature distance between real and fake samples in the discriminator feature space can stabilize the training of adversarial learning and contribute to the performance of a model as used in the FUNIT work. We take the feature before the last linear layer and performed spatial average pooling. Let the feature computing function as D_f. The feature matching loss is given by

$$\mathcal{L}_{\text{FM}}(G) = E_{x_s, x_c}[||D_f(x_s) - D_f(\bar{\boldsymbol{x_s}})||_1], \tag{6}$$

where $\bar{\boldsymbol{x_s}} = G(x_c, x_s)$.

Overall our training objective is given by,

$$\min_{D} \max_{G} \mathcal{L}_{\text{GAN}}(D, G) + \lambda_{\text{R}} \mathcal{L}_{\text{R}}(G) + \lambda_{\text{F}} \mathcal{L}_{\text{FM}}(G), \tag{7}$$

where λ_{R} and λ_{F} denote trade-off parameters for two losses. We set λ_{R} 0.1 and λ_{F} 1.0 in all experiments.

C Performance Metrics

Here, we describe the details of the evaluation metrics that we use to measure style faithfulness, content preservation, and human preference of the translation outputs.

Style faithfulness. We use the Frechet Inception Distance (FID) [51] to measure distance between the distribution of the translated images and the distribution of real unseen images, based on the InceptionV3 [52] network. We compute FID for each of the target class and report their mean (mFID).

Content preservation. An ideal translation should keep the structure of input content image unchanged. We measure the content preservation by comparing the body-part segmentation mask of a content image and that of a translated image. Since we do not have ground-truth body-part annotations, we estimate the body-part segmentation masks by using a DeeplabV3 [53] network trained on the Pascal body part dataset [54]. We note similar approaches are used in the other image synthesis prior works [2,16,24,36]. We obtain the body-part segmentation masks for both content and translated images. Then, we calculate pixel accuracy (PAcc) and mean intersection-over-union (mIoU) by treating the mask of the content image as the ground-truth annotation.

Human preference. Using Amazon Mechanical Turk (AMT), we perform a subjective visual test to gauge the quality of few-shot translation results. We conduct two studies: content preservation and style faithfulness. In the first study, each question contains the content image and the translation results from two competing methods, and the AMT worker is asked to choose which translation result better preserves the pose content from the content image. In the second study, each question contains the style image and the translation results from two competing methods, and the AMT worker is asked to choose which translation result rendering an object resembles more to the one in the style image. We generate 1000 questions per dataset per study. Each question is answered by 3 different AMT workers with a high approval rating. We use the preference score for evaluation. These are the counterpart of the automatic evaluation metrics described above.

D Experiments (Cont.)

Table 3: Ablation study on the Carnivores and Birds dataset. "ProjD" and "ResBlks+" represent the variant where we replace the mutli-class disciminator with the projection discriminator [45] and the variant where we use additional residual blocks in the decoder. "Ours w/o COCO" represents a baseline where COCO is removed.

Method	Carnivores			Birds		
	mFID↓	PAcc↑	mIou ↑	mFID↓	PAcc↑	mIou ↑
Ours w/o COCO w ProjD & ResBlks+	147.1	59.9	44.7	89.2	52.4	37.2
Ours w/o COCO w ResBlks+	114.3	65.7	51.3	75.0	51.9	37.3
Ours w/o COCO w ProjD	138.5	54.2	39.4	97.4	52.6	36.9
Ours w/o COCO	**99.6**	62.5	47.8	**68.8**	52.8	37.9
Ours	107.8	**66.5**	**52.1**	74.6	**53.3**	**38.3**

More ablation study. Table 3 shows the ablation of the proposed style encoder, projection discriminator [47] and additional residual blocks in decoder.

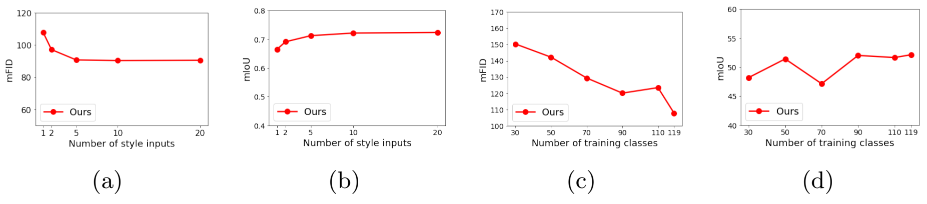

Fig. 16: (a)(b): mFID and mIoU scores with respect to varying number of style images respectively. (c)(d): mFID and mIoU with respect to varying number of training classes.

We find our full model better preserves content information and achieve better performance on the mFID metric. We also find that projection discriminator and additional residual blocks in the generator helps lower the mFID scores.

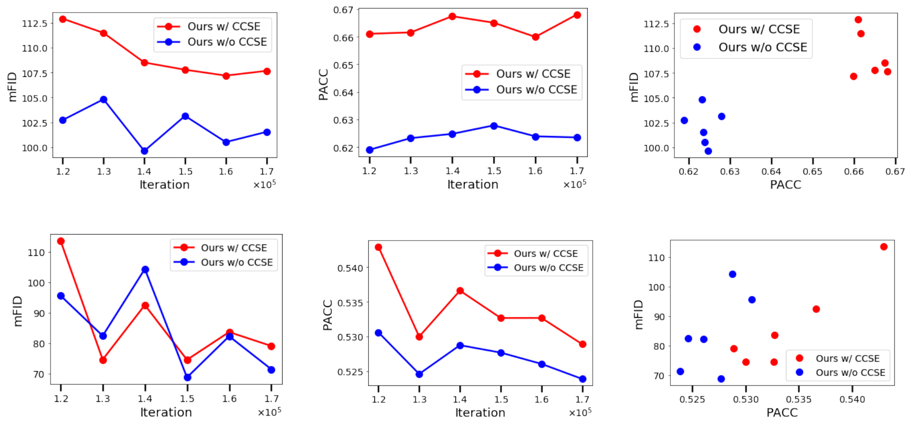

Fig. 17: Translation performance vs training iterations. Top row: Carnivorous dataset, bottom row: Bird dataset.

Effect on number of examples. We further investigate the relation between translation performance and number of input style images using the Carnivores dataset. Fig. 16a and 16b show that the translation performance of our method are positively correlated with the number of input style images.

Effect on number of source classes. Fig. 16c and 16d show the number of sources classes used in training versus the achieved mFID and mIoU score on the

Carnivores dataset. When the model sees more source classes during training, it renders a better few-shot image translation performance during testing.

Performance vs training time. Fig. 17 shows the training performance vs iterations on the Carnivores and Birds datasets. Both the mFID and PAcc scores improve as training proceeds for the Carnivores dataset. For the Birds dataset, the PAcc score degrades while the mFID improves. For both datasets, ours with COCO have better content preservation score.

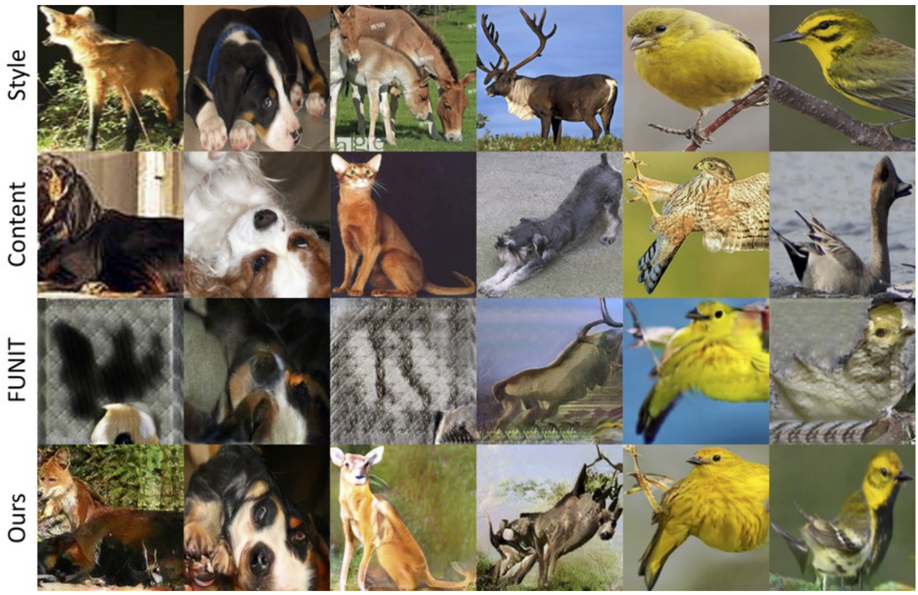

Fig. 18: Failure cases. Column 1 & 2 are from the Carnivores dataset. Column 3 & 4 are from the Mammals dataset. Column 5 & 6 are from the Birds dataset.

Failure cases. Fig. 18 illustrates several failure cases generated by our method. When the body part of the input content is hard to localize, the model generates incorrect results. Sometimes, the model generates hybrid classes.

Results on a well-aligned dataset. Fig. 19 shows the translation results on the Animal Faces dataset presented in FUNIT paper [10]. The images focus on animal face regions, thus they are well-aligned. We used the same training and test split like the original paper but performed translation on 256x256 image resolution. The mFID, PAcc, mIOU of FUNIT are 196.9, 0.505 and 0.344 respectively while ours are **106.8, 0.617, 0.459**. Even for the well-aligned dataset, we can see the advantage of using our architecture.

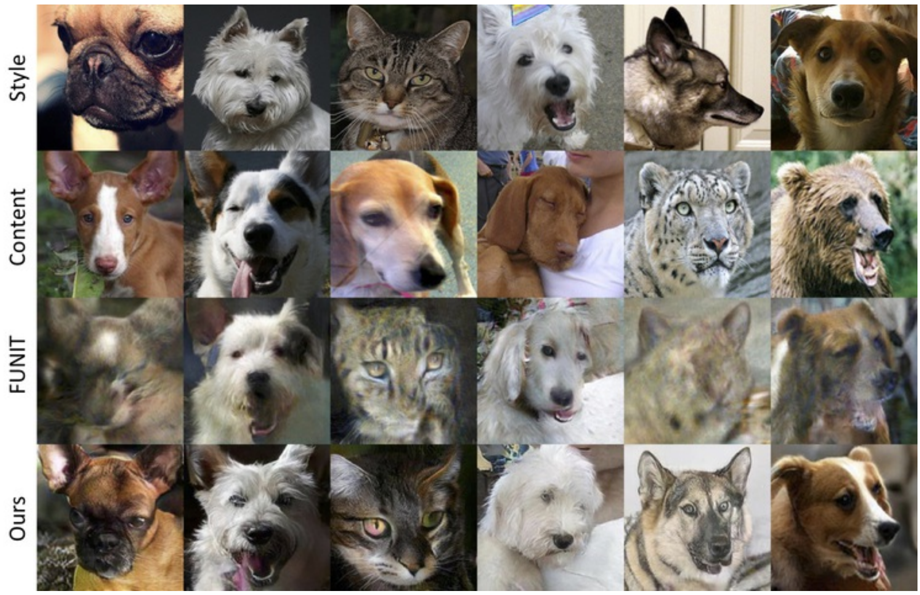

Fig. 19: Results on one-shot image-to-image translation for the Animal Faces dataset proposed in FUNIT [10] work.

Additional visual results. Fig. 21, 22, 23, and 20 show additional one-shot image translation results on the benchmark datasets. Our model can generate more photorealistic and more faithful translation outputs.

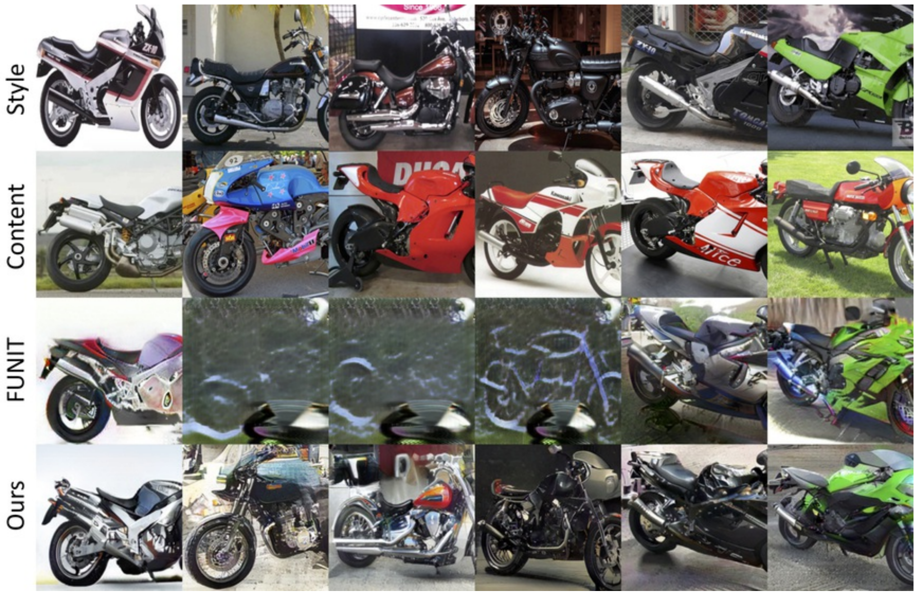

Fig. 20: One-shot image-to-image translation results on the Motorbike dataset.

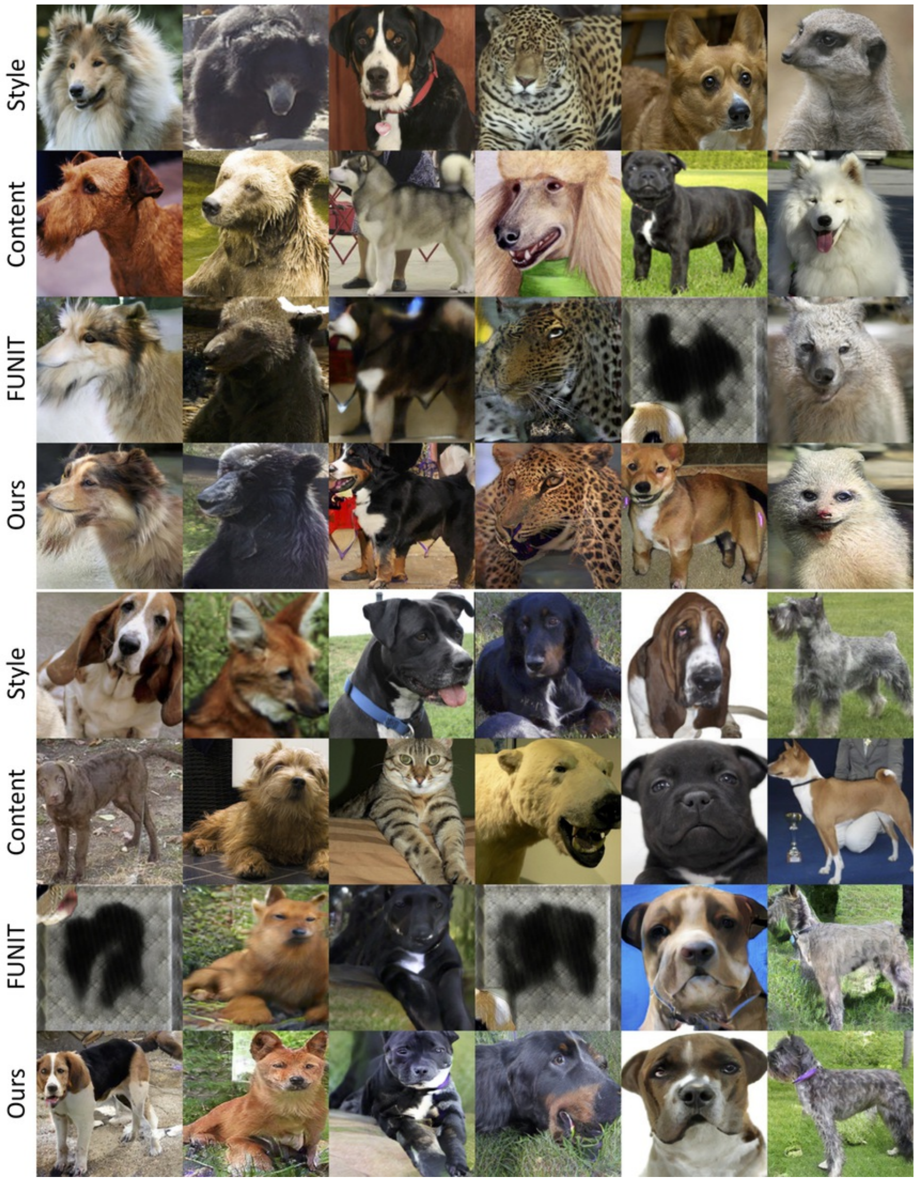

Fig. 21: One-shot image-to-image translation results on the Carnivores dataset.

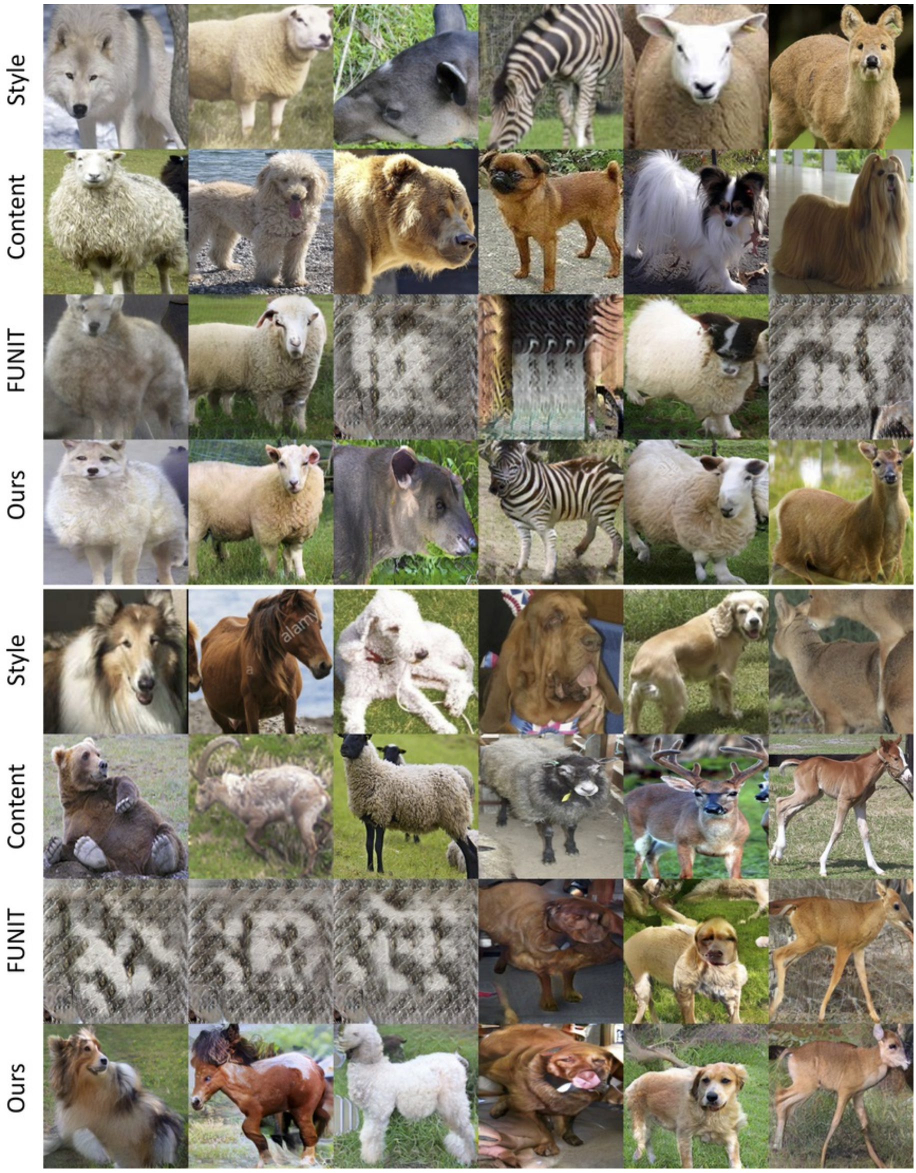

Fig. 22: One-shot image-to-image translation results on the Mammals dataset.

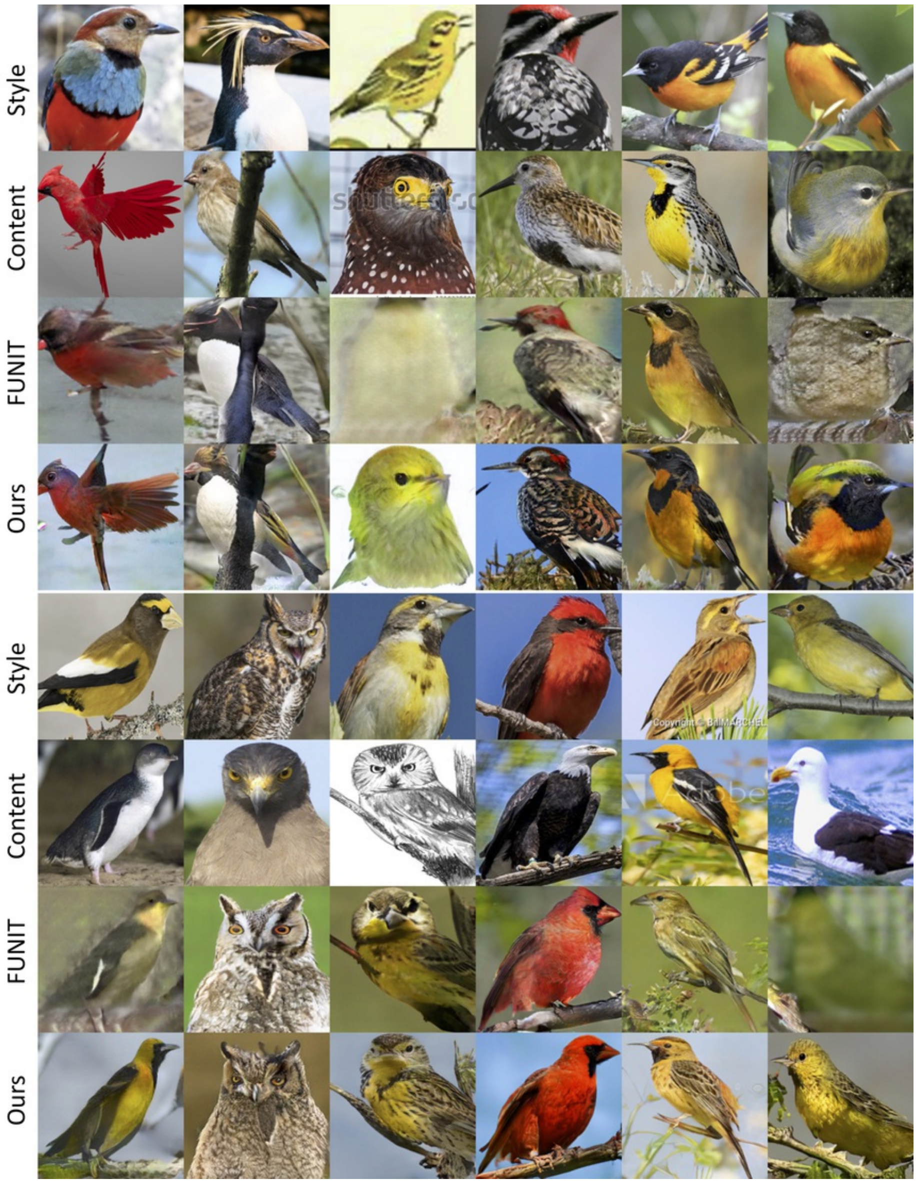

Fig. 23: One-shot image-to-image translation results on the Birds dataset.