Few-Shot Unsupervised Image-to-Image Translation

Ming-Yu Liu[1], Xun Huang[1,2], Arun Mallya[1], Tero Karras[1], Timo Aila[1], Jaakko Lehtinen[1,3], Jan Kautz[1]
[1]NVIDIA, [2]Cornell University, [3]Aalto University
{mingyul, xunh, amallya, tkarras, taila, jlehtinen, jkautz}@nvidia.com

Abstract

Unsupervised image-to-image translation methods learn to map images in a given class to an analogous image in a different class, drawing on unstructured (non-registered) datasets of images. While remarkably successful, current methods require access to many images in both source and destination classes at training time. We argue this greatly limits their use. Drawing inspiration from the human capability of picking up the essence of a novel object from a small number of examples and generalizing from there, we seek a few-shot, unsupervised image-to-image translation algorithm that works on previously unseen target classes that are specified, at test time, only by a few example images. Our model achieves this few-shot generation capability by coupling an adversarial training scheme with a novel network design. Through extensive experimental validation and comparisons to several baseline methods on benchmark datasets, we verify the effectiveness of the proposed framework. Our implementation and datasets are available at `https://github.com/NVlabs/FUNIT`.

1. Introduction

Humans are remarkably good at generalization. When given a picture of a previously unseen exotic animal, say, we can form a vivid mental picture of the same animal in a different pose, especially when we have encountered (images of) similar but different animals in that pose before. For example, a person seeing a standing tiger for the first time will have no trouble imagining what it will look lying down, given a lifetime of experience of other animals.

While recent unsupervised image-to-image translation algorithms are remarkably successful in transferring complex appearance changes across image classes [30, 46, 29, 25, 55, 52], the capability to generalize from few samples of a new class based on prior knowledge is entirely beyond their reach. Concretely, they need large training sets over all classes of images they are to perform translation on, i.e., they do not support few-shot generalization.

As an attempt to bridge the gap between human and machine imagination capability, we propose the Few-shot UNsupervised Image-to-image Translation (`FUNIT`) framework, aiming at learning an image-to-image translation model for mapping an image of a source class to an analogous image of a target class by leveraging few images of the target class given at test time. The model is never shown images of the target class during training but is asked to generate some of them at test time. To proceed, we first hypothesize that the few-shot generation capability of humans develops from their past visual experiences—a person can better imagine views of a new object if the person has seen many more different object classes in the past. Based on the hypothesis, we train our `FUNIT` model using a dataset containing images of many different object classes for simulating the past visual experiences. Specifically, we train the model to translate images from one class to another class by leveraging few example images of the another class. We hypothesize that by learning to extract appearance patterns from the few example images for the translation task, the model learns a generalizable appearance pattern extractor that can be applied to images of unseen classes at test time for the few-shot image-to-image translation task. In the experiment section, we give empirical evidence that the few-shot translation performance improves as the number of classes in the training set increases.

Our framework is based on Generative Adversarial Networks (GAN) [14]. We show that by coupling an adversarial training scheme with a novel network design we achieve the desired few-shot unsupervised image-to-image translation capability. Through extensive experimental validation on three datasets, including comparisons to several baseline methods using a variety of performance metrics, we verify the effectiveness of our proposed framework. In addition, we show the proposed framework can be applied to the few-shot image classification task. By training a classifier on the images generated by our model for the few-shot classes, we are able to outperform a state-of-the-art few-shot classification method that is based on feature hallucination.

2. Related Work

Unsupervised/unpaired image-to-image translation aims at learning a conditional image generation function that can map an input image of a source class to an ana-

Given a few target class this architecture translate content to target image which uses adversarial mechanism to train.

Only couple of target images are sufficient for training GAN that learn in an unsupervised way to translate source image to target image.

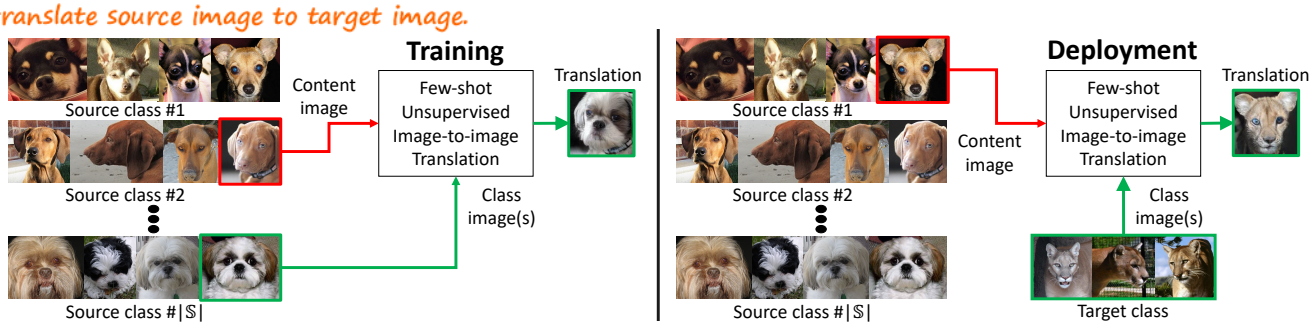


Figure 1. **Training.** The training set consists of images of various object classes (source classes). We train a model to translate images between these source object classes. **Deployment.** We show our trained model very few images of the target class, which is sufficient to translate images of source classes to analogous images of the target class even though the model has never seen a single image from the target class during training. Note that the FUNIT generator takes two inputs: 1) a content image and 2) a set of target class images. It aims to generate a translation of the input image that resembles images of the target class.

logues image of a target class without pair supervision. This problem is inherently ill-posed as it attempts to recover the joint distribution using samples from marginal distributions [29, 30]. To deal with the problem, existing works use additional constraints. For example, some works enforce the translation to preserve certain properties of the source data, such as pixel values [41], pixel gradients [5], semantic features [46], class labels [5], or pairwise sample distances [3]. There are works enforcing the cycle consistency constraint [52, 55, 25, 1, 56]. Several works use the shared/partially-shared latent space assumption [29, 30]/[19, 26]. Our work is based on the partially-shared latent space assumption but is designed for the few-shot unsupervised image-to-image translation task.

While capable of generating realistic translation outputs, existing unsupervised image-to-image translation models are limited in two aspects. First, they are sample inefficient, generating poor translation outputs if only few images are given at training time. Second, the learned models are limited for translating images between two classes. A trained model for one translation task cannot be directly reused for a new task despite similarity between the new task and the original task. For example, a husky-to-cat translation model can not be re-purposed for husky-to-tiger translation even though cat and tiger share a great similarity.

Recently, Benaim and Wolf [4] proposed an unsupervised image-to-image translation framework for partially addressing the first aspect. Specifically, they use a training dataset consisting of one source class image but many target class images to train a model for translating the *single* source class image to an analogous image of the target class. Our work differs from their work in several major ways. First, we assume many source class images but few target class images. Moreover, we assume that the few target class images are only available at test time and can be from many different object classes.

Multi-class unsupervised image-to-image translation [8, 2, 20] extends the unsupervised image-to-image translation methods to multiple classes. Our work is similar to these methods in the sense that our training dataset consists of images of multiple classes. But instead of translating images among *seen* classes, we focus on translating images of seen classes to analogous images of previously *unseen* classes.

Few-shot classification. Unlike few-shot image-to-image translation, the task of learning classifiers for novel classes using few examples is a long-studied problem. Early works use generative models of appearance that share priors across classes in a hierarchical manner [11, 39]. More recent works focus on using meta-learning to quickly adapt models to novel tasks [12, 35, 38, 34]. These methods learn better optimization strategies for training, so that the performance upon seeing only few examples is improved. Another set of works focus on learning image embeddings that are better suited for few-shot learning [49, 43, 44]. Several recent works propose augmenting the training set for the few-shot classification task by generating new feature vectors corresponding to novel classes [10, 15, 51]. Our work is designed for few-shot unsupervised image-to-image translation. However, it can be applied to few-shot classification, as shown in the experiments section.

3. Few-shot Unsupervised Image Translation

The proposed FUNIT framework aims at mapping an image of a source class to an analogous image of an unseen target class by leveraging a few target class images that are made available at test time. To train FUNIT, we use images from a set of object classes (*e.g.* images of various animal species), called the source classes. We do not assume existence of paired images between any two classes (*i.e.* no two animals of different species are at exactly the same pose). We use the source class images to train a multi-class unsupervised image-to-image translation model. During testing, we provide the model few images from a novel object class,

FUNIT framework works as follows:
1. During training they use set of source classes (various animal species) and a few target images.
2. They use adversarial mechanism to train Generator that leverage the few target images to translate any source class image to analogous images of the target class

Also, if we give the same model few images from a different object class (target), it has to translate any source class images to analogous images of the different novel object class.

One important difference in compare to conditional image generators in existing unsupervised setting is, here Generator G simultaneously takes a content image x and a set of K class images as input and produce output image x' (translated image).

called the target class. The model has to leverage the few target images to translate any source class image to analogous images of the target class. When we provide the same model few images from a different novel object class, it has to translate any source class images to analogous images of the different novel object class.

Our framework consists of a conditional image generator G and a multi-task adversarial discriminator D. Unlike the conditional image generators in existing unsupervised image-to-image translation frameworks [55, 29], which take one image as input, our generator G simultaneously takes *a content image* $\mathbf{x}$ and *a set of K class images* $\{\mathbf{y}_1, ..., \mathbf{y}_K\}$ as input and produce the output image $\bar{\mathbf{x}}$ via

Take both input, set of target class image and a source image which finally give x' as translated image.

$$\bar{\mathbf{x}} = G(\mathbf{x}, \{\mathbf{y}_1, ..., \mathbf{y}_K\}). \quad (1)$$

We assume the content image belongs to object class c_x while each of the K class images belong to object class c_y. In general, K is a small number and c_x is different from c_y. We will refer G as the few-shot image translator.

As shown in Figure 1, G maps an input content image $\mathbf{x}$ to an output image $\bar{\mathbf{x}}$, such that $\bar{\mathbf{x}}$ looks like an image belonging to object class c_y, and $\bar{\mathbf{x}}$ and $\mathbf{x}$ share structural similarity. Let $\mathbb{S}$ and $\mathbb{T}$ denote the set of source classes and the set of target classes, respectively. During training, G learns to translate images between two randomly sampled source classes $c_x, c_y \in \mathbb{S}$ with $c_x \neq c_y$. At test time, G takes a few images from an unseen target class $c \in \mathbb{T}$ as the class images, and maps an image sampled from any of the source classes to an analogous image of the target class c.

Next, we discuss the network design and learning. More details are given in Appendix A.

3.1. Few-shot Image Translator

The few-shot image translator G consists of a content encoder E_x, a class encoder E_y, and a decoder F_x. The content encoder is made of several 2D convolutional layers followed by several residual blocks [16, 22]. It maps the input content image $\mathbf{x}$ to a content latent code $\mathbf{z}_x$, which is a spatial feature map. The class encoder consists of several 2D convolutional layers followed by a mean operation along the sample axis. Specifically, it first maps each of the K individual class images $\{\mathbf{y}_1, ..., \mathbf{y}_K\}$ to an intermediate latent vector and then computes the mean of the intermediate latent vectors to obtain the final class latent code $\mathbf{z}_y$.

The decoder consists of several adaptive instance normalization (AdaIN) residual blocks [19] followed by a couple of upscale convolutional layers. The AdaIN residual block is a residual block using the AdaIN [18] as the normalization layer. For each sample, AdaIN first normalizes the activations of a sample in each channel to have a zero mean and unit variance. It then scales the activations using a learned affine transformation consisting of a set of scalars and biases. Note that the affine transformation is spatially invariant and hence can only be used to obtain global appearance information. The affine transformation parameters are adaptively computed using $\mathbf{z}_y$ via a two-layer fully connected network. With E_x, E_y, and F_x, (1) becomes

$$\bar{\mathbf{x}} = F_x\big(\mathbf{z}_x, \mathbf{z}_y\big) = F_x\big(E_x(\mathbf{x}), E_y(\{\mathbf{y}_1, ..., \mathbf{y}_K\})\big). \quad (2)$$

By using this translator design, we aim at extracting class-invariant latent representation (*e.g.*, object pose) using the content encoder and extracting class-specific latent representation (*e.g.*, object appearance) using the class encoder. By feeding the class latent code to the decoder via the AdaIN layers, we let the class images control the global look (*e.g.*, object appearance), while the content image determines the local structure (*e.g.*, locations of eyes).

At training time, the class encoder learns to extract class-specific latent representation from the images of the source classes. At test time, this generalizes to images of previously unseen classes. In the experiment section, we show that the generalization capability depends on the number of source object classes seen during training. When G is trained with more source classes (*e.g.*, more species of animals), it has a better few-shot image translation performance (*e.g.*, better in translating husky to mountain lion).

3.2. Multi-task Adversarial Discriminator

Our discriminator D is trained by solving multiple adversarial classification tasks simultaneously. Each of the tasks is a binary classification task determining whether an input image is a real image of the source class or a translation output coming from G. As there are $|\mathbb{S}|$ source classes, D produces $|\mathbb{S}|$ outputs. When updating D for a real image of source class c_x, we penalize D if its c_xth output is false. For a translation output yielding a fake image of source class c_x, we penalize D if its c_xth output is positive. We do not penalize D for not predicting false for images of other classes ($\mathbb{S} \backslash \{c_x\}$). When updating G, we only penalize G if the c_xth output of D is false. We empirically find this discriminator works better than a discriminator trained by solving a much harder $|\mathbb{S}|$-class classification problem.

3.3. Learning

We train the proposed `FUNIT` framework by solving a minimax optimization problem given by

$$\min_D \max_G \mathcal{L}_{\text{GAN}}(D, G) + \lambda_{\text{R}} \mathcal{L}_{\text{R}}(G) + \lambda_{\text{F}} \mathcal{L}_{\text{FM}}(G) \quad (3)$$

where $\mathcal{L}_{\text{GAN}}$, $\mathcal{L}_{\text{R}}$, and $\mathcal{L}_{\text{F}}$ are the GAN loss, the content image reconstruction loss, and the feature matching loss. The GAN loss is a conditional one given by

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, D) =& E_{\mathbf{x}}\left[-\log D^{c_x}(\mathbf{x})\right] + \\ & E_{\mathbf{x}, \{\mathbf{y}_1, ..., \mathbf{y}_K\}}[\log\left(1 - D^{c_y}\left(\bar{\mathbf{x}}\right)\right] \end{aligned} \quad (4)$$

	Setting	Top1-all ↑	Top5-all ↑	Top1-test ↑	Top5-test ↑	DIPD ↓	IS-all ↑	IS-test ↑	mFID ↓
Animal Faces	CycleGAN-Unfair-20	28.97	47.88	38.32	71.82	1.615	10.48	7.43	197.13
	UNIT-Unfair-20	22.78	43.55	35.73	70.89	1.504	12.14	6.86	197.13
	MUNIT-Unfair-20	38.61	62.94	53.90	84.00	1.700	10.20	7.59	158.93
	StarGAN-Unfair-1	2.56	10.50	9.07	32.55	1.311	10.49	5.17	201.58
	StarGAN-Unfair-5	12.99	35.56	25.40	60.64	1.514	7.46	6.10	204.05
	StarGAN-Unfair-10	20.26	45.51	30.26	68.78	1.559	7.39	5.83	208.60
	StarGAN-Unfair-15	20.47	46.46	34.90	71.11	1.558	7.20	5.58	204.13
	StarGAN-Unfair-20	24.71	48.92	35.23	73.75	1.549	8.57	6.21	198.07
	StarGAN-Fair-1	0.56	3.46	4.41	20.03	1.368	7.83	3.71	228.74
	StarGAN-Fair-5	0.60	3.56	4.38	20.12	1.368	7.80	3.72	235.66
	StarGAN-Fair-10	0.60	3.40	4.30	20.00	1.368	7.84	3.71	241.77
	StarGAN-Fair-15	0.62	3.49	4.28	20.24	1.368	7.82	3.72	228.42
	StarGAN-Fair-20	0.62	3.45	4.41	20.00	1.368	7.83	3.72	228.57
	FUNIT-1	17.07	54.11	46.72	82.36	1.364	22.18	10.04	93.03
	FUNIT-5	33.29	78.19	68.68	96.05	1.320	22.56	13.33	70.24
	FUNIT-10	37.00	82.20	72.18	97.37	1.311	22.49	14.12	67.35
	FUNIT-15	38.83	83.57	73.45	97.77	1.308	22.41	14.55	66.58
	FUNIT-20	**39.10**	**84.39**	**73.69**	**97.96**	**1.307**	**22.54**	**14.82**	**66.14**
North American Birds	CycleGAN-Unfair-20	9.24	22.37	19.46	42.56	1.488	25.28	7.11	215.30
	UNIT-Unfair-20	7.01	18.31	16.66	37.14	1.417	28.28	7.57	203.83
	MUNIT-Unfair-20	23.12	41.41	38.76	62.71	1.656	24.76	9.66	198.55
	StarGAN-Unfair-1	0.92	3.83	3.98	13.73	1.491	14.80	4.10	266.26
	StarGAN-Unfair-5	2.54	8.94	8.82	23.98	1.574	13.84	4.21	270.12
	StarGAN-Unfair-10	4.26	13.28	12.03	32.02	1.571	15.03	4.09	278.94
	StarGAN-Unfair-15	3.70	11.74	12.90	31.62	1.509	18.61	5.25	252.80
	StarGAN-Unfair-20	5.38	16.02	13.95	33.96	1.544	18.94	5.24	260.04
	StarGAN-Fair-1	0.24	1.17	0.97	4.84	1.423	13.73	4.83	244.65
	StarGAN-Fair-5	0.22	1.07	1.00	4.86	1.423	13.72	4.82	244.40
	StarGAN-Fair-10	0.24	1.13	1.03	4.90	1.423	13.72	4.83	244.55
	StarGAN-Fair-15	0.23	1.05	1.04	4.90	1.423	13.72	4.81	244.80
	StarGAN-Fair-20	0.23	1.08	1.00	4.86	1.423	13.75	4.82	244.71
	FUNIT-1	11.17	34.38	30.86	60.19	1.342	67.17	17.16	113.53
	FUNIT-5	20.24	51.61	45.40	75.75	1.296	74.81	22.37	99.72
	FUNIT-10	22.45	54.89	48.24	77.66	1.289	75.40	23.60	98.75
	FUNIT-15	23.18	55.63	49.01	78.70	1.287	**76.44**	23.86	98.16
	FUNIT-20	**23.50**	**56.37**	**49.81**	**78.89**	**1.286**	76.42	**24.00**	**97.94**

Table 1. Performance comparison with the fair and unfair baselines. ↑ means larger numbers are better, ↓ means smaller numbers are better.

The superscript attached to D denotes the object class; the loss is computed only using the corresponding binary prediction score of the class.

The content reconstruction loss helps G learn a translation model. Specifically, when using the same image for both the input content image and the input class image (in this case $K = 1$), the loss encourages G to generate an output image identical to the input

$$\mathcal{L}_{\text{R}}(G) = E_{\mathbf{x}}\left[||\mathbf{x} - G(\mathbf{x}, \{\mathbf{x}\})||_1^1\right]. \tag{5}$$

The feature matching loss regularizes the training. We first construct a feature extractor, referred to as D_f, by removing the last (prediction) layer from D. We then use D_f to extract features from the translation output $\bar{\mathbf{x}}$ and the class images $\{\mathbf{y}_1, ..., \mathbf{y}_K\}$ and minimize

$$\mathcal{L}_{\text{F}}(G) = E_{\mathbf{x},\{\mathbf{y}_1,...,\mathbf{y}_K\}}\left[||D_f(\bar{\mathbf{x}})) - \sum_k \frac{D_f(\mathbf{y}_k)}{K}||_1^1\right]. \tag{6}$$

Both of the content reconstruction loss and the feature matching loss are not new topics to image-to-image translation [29, 19, 50, 37]. Our contribution is in extending their use to the more challenging and novel few-shot unsupervised image-to-image translation setting.

4. Experiments

Implementation. We set $\lambda_{\text{R}} = 0.1$ and $\lambda_{\text{F}} = 1$. We optimize (3) using RMSProp with learning rate 0.0001. We use the hinge version of GAN loss [28, 33, 53, 6] and the real gradient penalty regularization proposed by Mescheder *et al*. [32]. The final generator is a historical average version of the intermediate generators [23] where the update weight is 0.001. We train the FUNIT model using $K = 1$ since we desire it to work well even when *only one* target class image is available at test time. In the experiments, we evaluate its performance under $K = 1, 5, 10, 15, 20$. Each training batch consists of 64 content images, which are evenly distributed on 8 V100 GPUs in an NVIDIA DGX1 machine.

Datasets. We use the following datasets for experiments.

- *Animal Faces.* We build this dataset using images from the 149 carnivorous animal classes in ImageNet [9]. We

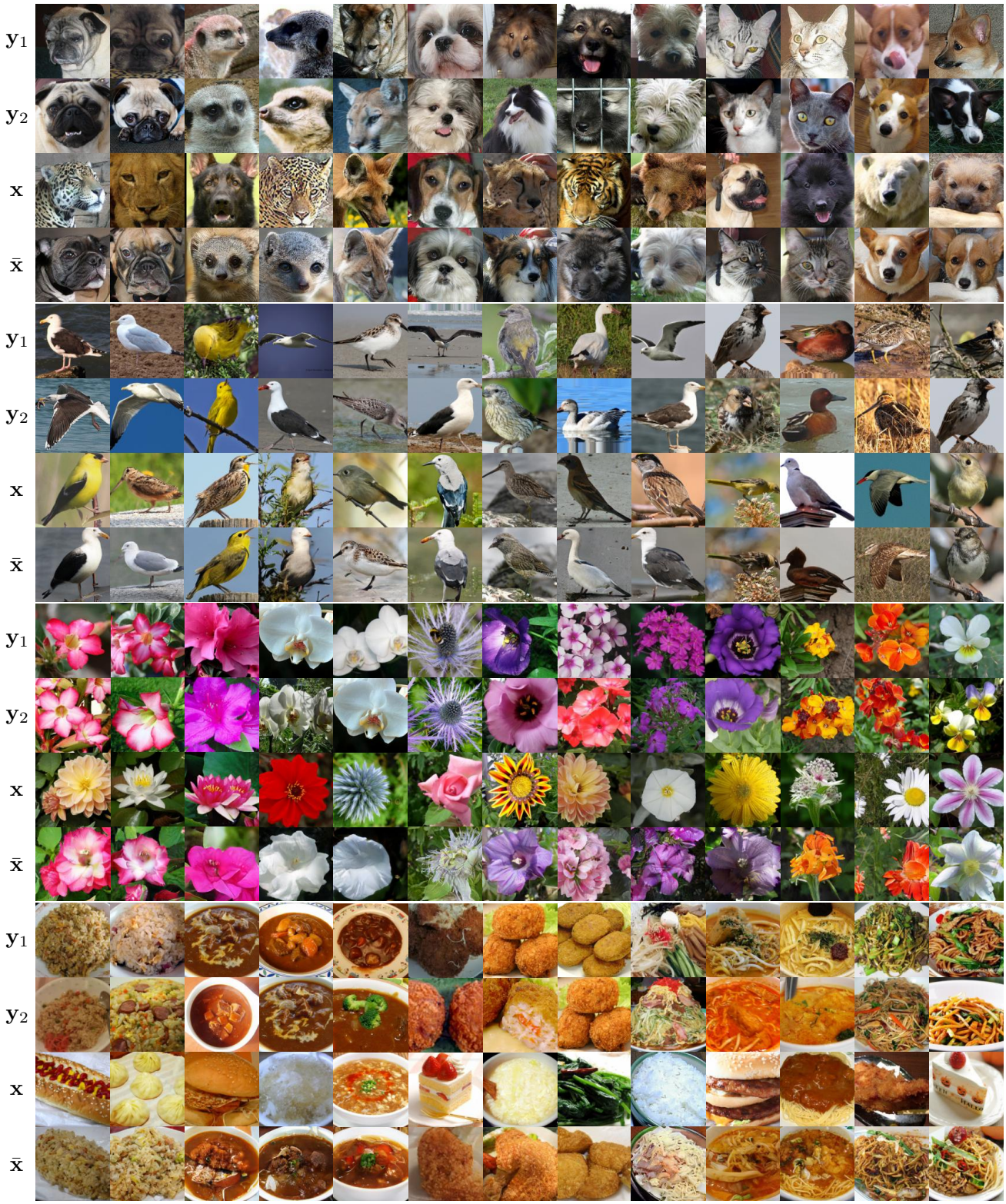


Figure 2. Visualization of the *few-shot* unsupervised image-to-image translation results. The results are computed using the FUNIT-5 model. From top to bottom, we have the results from the animal face, bird, flower, and food datasets. We train one model for each dataset. For each example, we visualize 2 out of 5 randomly sampled class images $\mathbf{y}_1\mathbf{y}_2$, the input content image $\mathbf{x}$, and the translation output $\bar{\mathbf{x}}$. The results show that FUNIT generate plausible translation outputs under the difficult few-shot setting where the models see no images from any of the target classes during training. We note that the objects in the output images have similar poses to the inputs.

first manually label bounding boxes of 10000 carnivorous animal faces in the images. We then train a Faster RCNN [13] to detect animal faces in the images. We only use the bounding boxes with high detection scores. This renders a set of 117574 animal faces. We split the classes into a source class set and a target class set, which contains 119 and 30 animal classes, respectively.

- *Birds* [48]. 48527 images of 555 North American bird species. 444 species are used for the source class set and 111 species are used for the target class set.
- *Flowers* [36]. 8189 images from 102 species. The source and target sets have 85 and 17 species, respectively.
- *Foods* [24]. 31395 images from 256 kinds of food. The source and target set have 224 and 32 kinds, respectively.

Baselines. Depending on whether images of the target class are available during training, we define two sets of baselines: *fair* (unavailable) and *unfair* (available).

- *Fair.* This is the setting of the proposed `FUNIT` framework. As none of the prior unsupervised image-to-image translation methods are designed for the setting, we build a baseline by extending the StarGAN method [8], which is the state of the art for multi-class unsupervised image-to-image translation. We train a StarGAN model purely using source class images. During testing, given K images of a target class, we compute the average VGG [42] Conv5 features for the K images and compute its cosine distance to the average VGG Conv5 feature for the images of each source class. We then compute the class association vector by applying softmax to the cosine distances. We use the class association vector as input to the StarGAN model (substituting the one-hot class association vector input) for generating images of unseen target classes. The baseline method is designed with the assumption that the class association scores could encode how an unseen target object class is related to each of the source classes, which can be used for few-shot generation. We denote this baseline `StarGAN-Fair-K`.

- *Unfair.* These baselines include target class images in the training. We vary the number of available images (K) per target class from 1 to 20 and train various unsupervised image-to-image translation models. We denote the StarGAN model that is trained with K images per target class as `StarGAN-Unfair-K`. We also train several state-of-the-art two-domain translation models including CycleGAN [55], UNIT [29], and MUNIT [19]. For them, we treat images of the source classes as the first domain and images of one target class as the second domain. This results in $|\mathbb{T}|$ unsupervised image-to-image translation models per dataset per two-class baseline. We label these baselines as `CycleGAN-Unfair-K`, `UNIT-Unfair-K`, and `MUNIT-Unfair-K`.

For the baseline methods, we use the source code and default parameter settings provided by the authors.

Figure 3. Visual comparison of few-shot image-to-image translation performance. From left to right, the columns are input content images $\mathbf{x}$, the two input target class images $\mathbf{y}_1$ $\mathbf{y}_2$, translation results from the unfair `StarGAN` baseline, translation results from the fair `StarGAN` baseline, and results from our framework.

Evaluation protocol. We use a randomly sampled 25000 images from the source classes as the content images. We then translate them to each target class by randomly sample K images of the target class. This produces $|\mathbb{T}|$ sets of images for each competing approach and they are used for evaluation. We use the same K images for each content image for all the competing approaches. We test a range of K values, including 1, 5, 10, 15, and 20.

Performance metrics. We use several criteria for performance comparison. First, we measure whether translations resemble images of the target class. Second, we examine whether class-invariant content are preserved during translation. Third, we quantify photorealism of output images. Finally, we measure whether the model can be used to generate the image distribution of a target class. More details on the performance metrics are given in Appendix B.

- *Translation accuracy* measure whether a translation output belongs to the target class. We use two Inception-V3 [45] classifiers. One classifier is trained using both of the source and target classes (denoted as *all*), while the other is trained using the target classes along (denoted as *test*). We report both Top1 and Top5 accuracies.

- *Content preservation* is based on a variant of perceptual

distance [22, 54], called the domain-invariant perceptual distance (DIPD) [19]. The distance is given by L2 distance between two normalized VGG [42] Conv5 features, which is more invariant against domain change [19].

- *Photorealism.* This is measured by the inception scores (IS) [40]. We report inception scores using the two inception classifiers trained for measuring translation accuracy, denoted by *all* and *test*, respectively.
- *Distribution matching* is based on Fréchet Inception Distance (FID) [17]. We compute FID for each of the $|\mathbb{T}|$ target object classes and report their mean FID (mFID).

Main results. As shown in Table 1, the proposed FUNIT framework outperforms the baselines for the few-shot unsupervised image-to-image translation task on all the performance metrics for both the Animal Faces and North American Birds datasets. FUNIT achieves 82.36 and 96.05 Top-5 (*test*) accuracy for the 1-shot and 5-shot settings, respectively, on the Animal Face dataset, and 60.19 and 75.75 on the North American Birds dataset. They are all significantly better than those achieved by the corresponding fair baselines. Similar trends can be found for the domain invariant perceptual distance, inception score, and Fréchet inception distance. Moreover, with just 5 shots, FUNIT outperforms all the unfair baselines under 20-shot settings. Note that for the results of CycleGAN-Unfair-20, UNIT-Unfair-20, and MUNIT-Unfair-20 are from $|\mathbb{T}|$ image-to-image translation networks, while our method is from a single translation network.

The table also shows that the performance of the proposed FUNIT model is positively correlated with number of available target images K at test time. A larger K leads to improvements across all the metrics, and the largest performance boost comes from $K = 1$ to $K = 5$. The StarGAN-Fair baseline does not exhibit a similar trend.

In Figure 2, we visualize the few-shot translation results computed by FUNIT-5. The results show that the FUNIT model can successfully translate images of source classes to analogous images of novel classes. The poses of the object in the input content image $\mathbf{x}$ and the corresponding output image $\bar{\mathbf{x}}$ remain largely the same. The output images are photorealistic and resemble images from the target classes. More results are given in Appendix I.

In Figure 3, we provide a visual comparison. As the baselines are not designed for the few-shot image translation setting, they failed in the challenging translation task. They either generate images with a large amount of artifacts or just output the input content image. On the other hand, FUNIT generates high-quality image translation outputs.

User study. To compare the photorealism and faithfulness of the translation outputs, we perform human evaluation using the Amazon Mechanical Turk (AMT) platform. Specifically, we give the workers a target class image and two translation outputs from different methods [50, 19] and ask them to choose the output image that resembles more the target class image. The workers are given unlimited time to make the selection. We use both the Animal Faces and North American Birds datasets. For each comparison, we randomly generate 500 questions and each question is answered by 5 different workers. For quality control, a worker must have a lifetime task approval rate grater than 98% to be able to participate in the evaluation.

Setting	Animal	Birds
FUNIT-5 vs. StarGAN-Fair-5	86.08	82.56
FUNIT-5 vs. StarGAN-Unfair-20	86.00	84.48
FUNIT-5 vs. CycleGAN-Unfair-20	71.68	77.76
FUNIT-5 vs. UNIT-Unfair-20	77.84	77.96
FUNIT-5 vs. MUNIT-Unfair-20	83.56	79.64

Table 2. User preference score. The numbers indicate the percentage of users favors results generated by the proposed method over those generated by the competing method.

# of generated samples N	Animal Face		North American Birds	
	S&H [15]	FUNIT	S&H [15]	FUNIT
0	38.76		30.38	
10	40.51	**42.05**	31.77	**33.41**
50	40.24	**42.22**	31.66	**33.64**
100	40.76	**42.14**	32.12	**34.39**

Table 3. Few-shot classification accuracies averaged over 5 splits.

According to Table 2, the human subjects consider the translation outputs generated by the proposed method under the 5-shot setting (FUNIT-5) much more similar to the target class images than those generate by the fair baseline under the same setting (StarGAN-Fair-5). Even when compared with the results of unfair baselines that have access to 20 images per target class at training time, our translation results are still considered to be much more faithful.

Number of source classes in the training set. In Figure 4, we analyze the performance versus varying number of source classes in the training set under the one-shot setting (FUNIT-1), using the animal dataset. We plot the curves by varying the number from 69 to 119 classes with an interval of 10. As shown, the performance is positively correlated with the number of object classes in terms of translation accuracy, image quality, and distribution matching. The domain-invariant perceptual distance remains flat. This shows that a FUNIT model that sees more object classes (larger diversity) during training performs better during testing. A similar trend is observed for the bird dataset, which is given in Appendix C.

Parameter analysis and ablation study. We analyze the impact of the individual terms in our objective function and find all of them are essential. Particularly, the content reconstruction loss trades translation accuracy for content preservation score. The details are given in Appendix D.

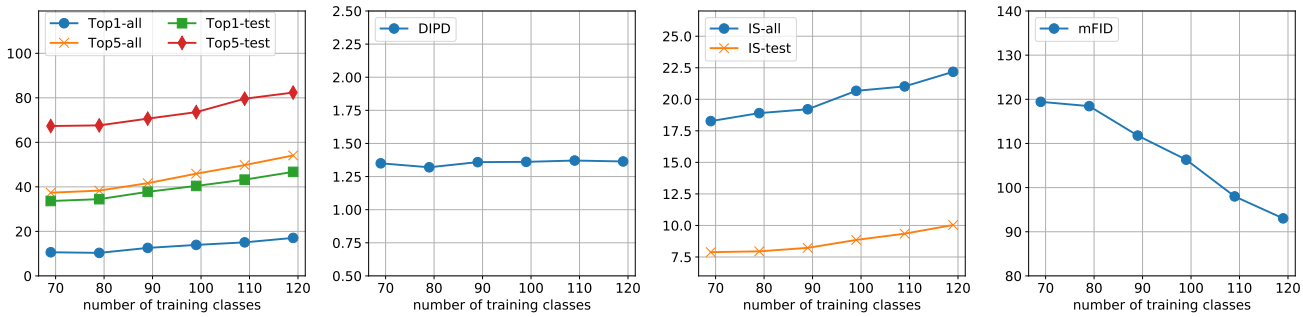


Figure 4. Few-shot image translation performance vs. number of object classes seen during training on the Animal Faces dataset. The performance is positively correlated with number of source object classes seen during training.

Figure 5. Limitations of the proposed framework. When the appearance of a unseen object class is dramatically different to the appearances of the source classes, (*e.g.* flower and animal face). The proposed FUNIT framework fails to generate meaningful translation outputs.

Comparison with the AdaIN style transfer method. We train an AdaIN style transfer network [18] for the few-shot animal face translation task and compare the performance as shown in Appendix E. We find that while the style transfer network can change the textures of the input animals, it does not change their shapes. As a result, the translation outputs still resemble to the inputs.

Failure cases. Several failure cases of the proposed algorithm are visualized in Appendix F. They include generating hybrid objects, ignoring input content images, and ignoring input class images.

Latent interpolation. In Appendix G, we show interpolation results by keeping the content code fixed while interpolating the class code between two source class images. Interestingly, we find that by interpolating between two source classes (Siamese cat and Tiger) we can sometimes generate a target class (Tabby cat) that the model has never observed.

Few-shot translation for few-shot classification. We evaluate FUNIT for few-shot classification using the animal and bird datasets. Specifically, we use the trained FUNIT models to generate N (varying from 1, 50, to 100) images for each of the few-shot classes and use the generated images to train the classifiers. We find the classifiers trained with the FUNIT generated images consistently achieve better performance than the few-shot classification approach proposed of S&H by Hariharan *et al*. [15], which is based on feature hallucination and also has a controllable variable on sample number N. The results are shown in Table 3 and the experiment details are given in Appendix H.

5. Discussion and Future Work

We introduced the first few-shot unsupervised image-to-image translation framework. We showed that the few-shot generation performance is positively correlated with the number of object classes seen during training and also positively correlated with the number of target class shots provided during test time.

We provided empirical evidence that FUNIT can learn to translate an image of a source class to a corresponding image of an unseen object class by utilizing few example images of the unseen class made available at test time. Although achieving this new capability, FUNIT depends on several conditions to work: 1) whether the content encoder E_x can learn a class-invariant latent code $\mathbf{z}_x$, 2) whether the class encoder E_y can learn a class-specific latent code $\mathbf{z}_y$, and, most importantly, 3) whether the class encoder E_y can generalize to images of unseen object classes.

We observed these conditions are easy to meet when the novel classes are visually related to the source classes. However, when the appearance of novel object classes are dramatically different from those of the source classes, FUNIT fails to achieve translation as shown in Figure 5. In this case, FUNIT tends to generate color-changed versions of the input content images. This is undesirable but understandable as the appearance distribution has changed dramatically. Addressing this limitation is our future work.

References

[1] Amjad Almahairi, Sai Rajeswar, Alessandro Sordoni, Philip Bachman, and Aaron Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. *arXiv preprint arXiv:1802.10151*, 2018.

[2] Asha Anoosheh, Eirikur Agustsson, Radu Timofte, and Luc Van Gool. Combogan: Unrestrained scalability for image domain translation. *arXiv preprint arXiv:1712.06909*, 2017.

[3] Sagie Benaim and Lior Wolf. One-sided unsupervised domain mapping. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[4] Sagie Benaim and Lior Wolf. One-shot unsupervised cross domain translation. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[5] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations (ICLR)*, 2019.

[7] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[8] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[10] Mandar Dixit, Roland Kwitt, Marc Niethammer, and Nuno Vasconcelos. Aga: Attribute-guided augmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[11] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2006.

[12] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, 2017.

[13] Ross Girshick. Fast r-cnn. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[15] Bharath Hariharan and Ross B Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[17] 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 equilibrium. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[18] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[19] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. *European Conference on Computer Vision (ECCV)*, 2018.

[20] Le Hui, Xiang Li, Jiaxin Chen, Hongliang He, and Jian Yang. Unsupervised multi-domain image translation with domain-specific encoders/decoders. *arXiv preprint arXiv:1712.02050*, 2017.

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[22] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision (ECCV)*, 2016.

[23] 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 (ICLR)*, 2018.

[24] Yoshiyuki Kawano and Keiji Yanai. Automatic expansion of a food image dataset leveraging existing categories with domain adaptation. In *European Conference on Computer Vision (ECCV) Workshop*, 2014.

[25] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jungkwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017.

[26] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Kumar Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representation. In *European Conference on Computer Vision (ECCV)*, 2018.

[27] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

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

[29] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[30] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[31] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[32] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International Conference on Machine Learning (ICML)*, 2018.

[33] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[34] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *International Conference on Machine Learning (ICML)*, 2017.

[35] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[36] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.

[37] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[38] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[39] Ruslan Salakhutdinov, Joshua Tenenbaum, and Antonio Torralba. One-shot learning with a hierarchical nonparametric bayesian model. In *International Conference on Machine Learning (ICML) Workshop*, 2012.

[40] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[41] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russ Webb. Learning from simulated and unsupervised images through adversarial training. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[43] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[44] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[45] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[46] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. In *International Conference on Learning Representations (ICLR)*, 2017.

[47] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[48] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[49] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[50] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[51] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[52] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[53] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

[54] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[55] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[56] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

A. Network Architecture

The few-shot image translator consists of three sub-networks: a content encoder, a class encoder, and a decoder as visualized in Figure 6. The content encoder maps the input content image to a content latent code, which is a feature map. If the resolution of the input image is 128x128, the resolution of the feature map will be 16x16 since there are 3 stride-2 down-sampling operations. This feature map is designed to encode class-invariant content information. It should encode locations of the parts but not their class-specific appearances[1]. On the other hand, the class encoder maps a set of K class images to a class latent code, which is a vector and is aimed to be class-specific. It first maps each input class image to an intermediate latent code using

[1]For example, in the animal face translation task, it should encode locations of the ears but not their shape and color.

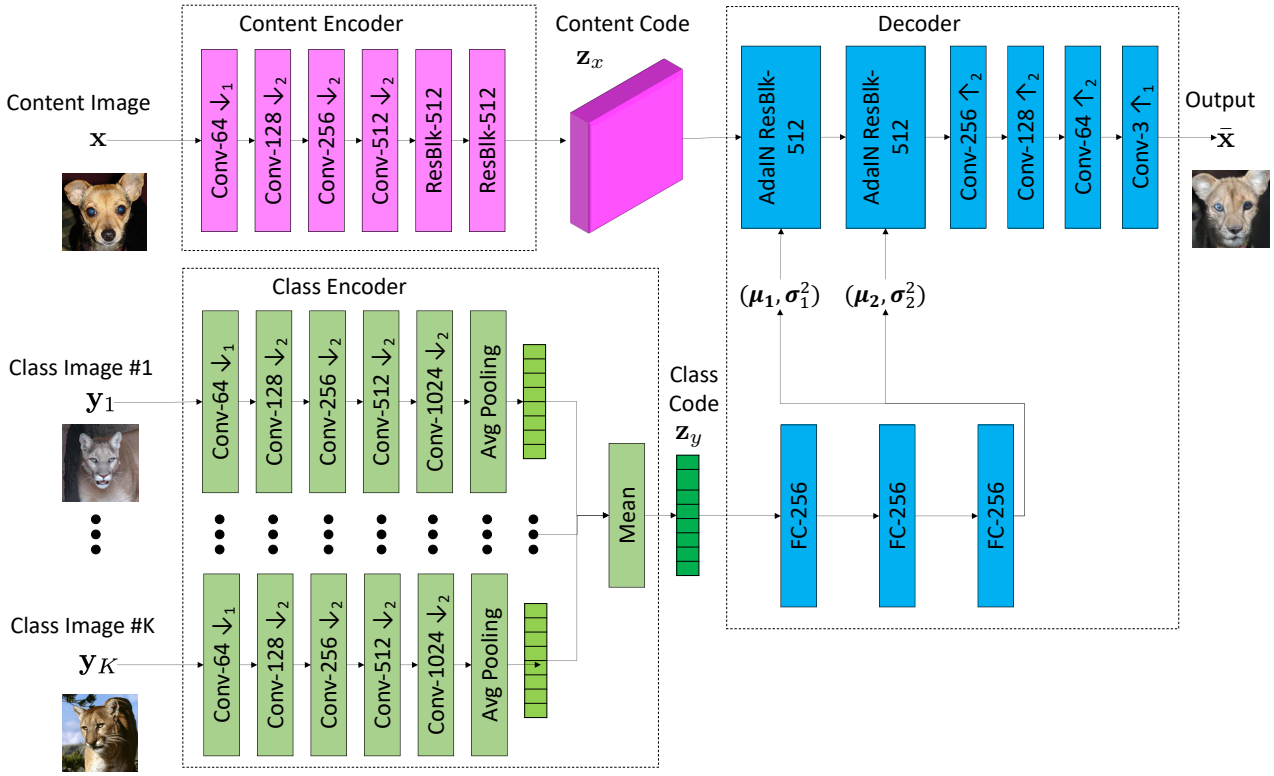


Figure 6. Visualization of the generator architecture. To generate a translation output $\bar{\boldsymbol{x}}$, the translator combines the class latent code $\boldsymbol{z}_y$ extracted from the class images $\boldsymbol{y}_1, ...\boldsymbol{y}_K$ with the content latent code $\boldsymbol{z}_x$ extracted from the input content image. Note that nonlinearity and normalization operations are not included in the visualization.

a VGG-like network. These latent vectors are then element-wise averaged to produce the final class latent code.

As shown in the figure, the decoder first decodes the class-specific latent code to a set of mean and variance vectors $(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)$ where $i = 1, 2$. These vectors are then used as the affine transformation parameters in the AdaIN residual blocks where $\boldsymbol{\sigma}_i^2$'s are the scaling factors and $\boldsymbol{\mu}_i$'s are the biases. For each residual block, the same affine transformation is applied to every spatial location in the feature map. It controls how the content latent code are decoded to generate the output image.

The number shown in each block in Figure 6 denotes the number of filters in the layer. The nonlinearity and normalization operations included in the network are excluded in the visualization. For the content encoder, each layer is followed by the instance normalization and the ReLU nonlinearity. For the class encoder, each layer is followed by the ReLU nonlinearity. For the decoder, except for the AdaIN residual blocks, each layer is followed by the instance normalization and the ReLU nonlinearity. We upscale the feature maps along each spatial dimension by a factor of 2 using nearest neighbor upsampling.

Our discriminator is a Patch GAN discriminator [21]. It utilizes the Leaky ReLU nonlinearity and employs no normalization. The discriminator consists of one convolutional layer followed by 10 activation first residual blocks [32]. The architecture is illustrated via the following chain of operations:

`Conv-64` $\rightarrow$ `ResBlk-128` $\rightarrow$ `ResBlk-128` $\rightarrow$ `AvePool2x2` $\rightarrow$ `ResBlk-256` $\rightarrow$ `ResBlk-256` $\rightarrow$ `AvePool2x2` $\rightarrow$ `ResBlk-512` $\rightarrow$ `ResBlk-512` $\rightarrow$ `AvePool2x2` $\rightarrow$ `ResBlk-1024` $\rightarrow$ `ResBlk-1024` $\rightarrow$ `AvePool2x2` $\rightarrow$ `ResBlk-1024` $\rightarrow$ `ResBlk-1024` $\rightarrow$ `Conv-`$||\mathbb{S}||$

where $||\mathbb{S}||$ is the number of source classes.

B. Performance Metrics

Translation accuracy. We use two Inception-V3 [45] classifiers to measure translation accuracy. The first classifier, denoted as *all*, is obtained by finetuning the ImageNet-pretrained Inception-V3 model on the task of classifying all the source and target object classes (*e.g.* all of the 149 classes for the Animal Faces dataset and all of the 555 classes for the North American Birds dataset). The second classifier (denoted as *test*) is obtained by finetuning the ImageNet-pretrained Inception-V3 model on the task of classifying the target object classes (*e.g.* 30 target classes

for the Animal Faces dataset and 111 target classes for the North American Birds dataset). We apply the classifiers to the translation output to see if they can recognize the output as an image of the target class. If yes, we denote it as a correct translation. We compare performance of competing models using both Top1 and Top5 accuracies. We thus have 4 evaluation metrics for translation accuracy: *Top1-all*, *Top5-all*, *Top1-test*, and *Top5-test*. An unsupervised image-to-image translation model with a higher accuracy is better. We note that similar evaluation protocols were used for comparing image-to-image translation models on the semantic label map to image translation task [21, 50, 7].

Content preservation. We quantify the content preservation performance using the domain-invariant perceptual distance (DIPD) [19]. The DIPD is a variant of perceptual distance [22, 54]. To compute the DIPD, we first extract the VGG [42] conv5 feature from the input content image as well as from the output translation image. We then apply the instance normalization [47] to the features, which will remove their mean and variance. This way, we can filter out much class-specific information in the features [18, 27] and focus on the class-invariant similarity. The DIPD is given by L2 distance between the instance normalized features.

Photorealism. We use the inception score (IS) [40], which is widely used for quantifying image generation performance. Let $p(t|\mathbf{y})$ be the distribution of class label t of the inception model over the output translation image $\mathbf{y}$. The inception score is given by

$$\text{IS}_C = \exp(\text{E}_{\mathbf{y}\sim p(\mathbf{y})}[\text{KL}(p(t|\mathbf{y})|p(t))]) \tag{7}$$

where $p(t) = \int_{\mathbf{y}} p(t|\mathbf{y})d\mathbf{y}$. It is argued in Salimans *et al*. [40] that the inception score is positively correlated with visual quality of neural network-generated images.

Distribution matching. The Frechet Inception Distance FID [17] is designed for measuring similarities between two sets of images. We use the activations from the last average pooling layer of the ImageNet-pretrained Inception-V3 [45] model as the feature vector of an image for computing FID. As we have $|\mathbb{T}|$ unseen classes, we translate source images to each of the $|\mathbb{T}|$ unseen classes and produce $|\mathbb{T}|$ sets of translation outputs. For each of the $|\mathbb{T}|$ sets of translation outputs, we compute the FID between the set to the corresponding set of ground truth images. This renders $|\mathbb{T}|$ FID scores. The average of the $|\mathbb{T}|$ FID scores is used as our final distribution matching performance metric, which is referred to as the mean FID (mFID).

C. Effect on Number of Source Classes

In the main paper, we show that the few-shot translation performance is positively correlated with number of source classes in the training set for the animal face translation task. In Figure 7, we show this is the same case for the bird translation task. Specifically, we report performance of the proposed model versus number of available source classes in the training set using the North American Birds dataset. We vary the number of source classes from 189, 222, 289, 333, 389, to 444 and plot the performance scores. We find the curves of the scores follow the same trend as those of the Animal Faces dataset shown in the main paper. When the model sees a larger number of source classes during training, it performs better during testing.

D. Ablation Study

In Table 4, we analyze impact of the content image reconstruction loss weight on the Animal Faces dataset. We find that a larger λ_R value leads to a smaller domain-invariant perceptual distance with the expense of a lower translation accuracy. The table shows that $\lambda_R = 0.1$ provides a good trade-off, and we used it as the default value throughout the paper. Interestingly, a very small weight value $\lambda_R = 0.01$ results in degrading performance on both content preservation and translation accuracy. This indicates that the content reconstruction loss help regularize the training.

Table 5 presents results of an ablation study analyzing impact of the loss terms in the proposed algorithm on the Animal Faces dataset. We find that removing the feature matching loss term resulting in a slightly degraded performance. But when removing the zero-centered gradient penalty, both content preservation and translation accuracy degrade a lot.

In Figure 8, we plot performance of the proposed model over training iterations on the one-shot setting (`FUNIT-1`). The translation accuracy, content preservation, image quality, and distribution matching scores improve with more iterations in general. The improvement is more dramatic in the early stage and slows down around 10000 iterations. We hence use 10000 iterations as the default parameter for reporting experiment results throughout the paper.

E. Comparison with AdaIN Style Transfer

In Figure 9, we compare the proposed method with the AdaIN style transfer method [18] for the few-shot animal face translation task. While the AdaIN style transfer method can change the textures of the input animals, it does not change their shapes. As a result, the translation outputs still resemble to the inputs in terms of appearances.

F. Failure Case

Figure 10 illustrates several failure cases of the proposed algorithm. They include generating hybrid classes, ignoring input content images, and ignoring input class images.

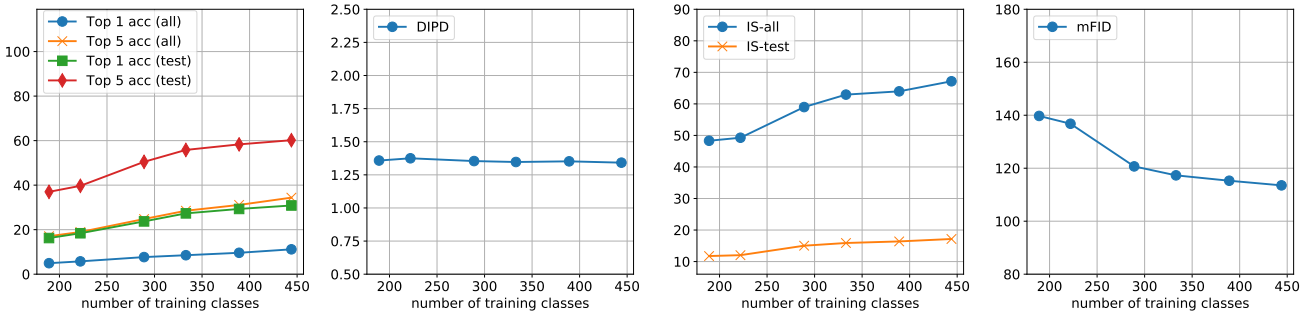


Figure 7. Few-shot image translation performance vs. number of object classes seen during training on the North American Birds dataset.

Setting	**Top1-all** ↑	**Top5-all** ↑	**Top1-test** ↑	**Top5-test** ↑	**DIPD** ↓	**IS-all** ↑	**IS-test** ↑	**mFID** ↓
$\lambda_R = 0.01$	16.02	52.30	45.52	81.68	1.370	21.80	9.73	94.98
$\lambda_R = 0.1$	**17.07**	**54.11**	**46.72**	**82.36**	1.364	22.18	**10.04**	**93.03**
$\lambda_R = 1$	16.60	52.05	45.62	81.77	1.346	**22.21**	9.81	94.23
$\lambda_R = 10$	13.04	44.32	39.06	75.81	**1.298**	20.48	8.90	108.71

Table 4. Parameter sensitivity analysis on the content image reconstruction loss weight, λ_R. ↑ means larger numbers are better, ↓ means smaller numbers are better. The value of 0.1 provides a good trade-off between content preservation and translation accuracy, which is used as the default value throughout the paper. We use the `FUNIT-1` model for this experiment.

Setting	**Top1-all** ↑	**Top5-all** ↑	**Top1-test** ↑	**Top5-test** ↑	**DIPD** ↓	**IS-all** ↑	**IS-test** ↑	**mFID** ↓
~~FM~~	15.33	52.98	46.33	**82.43**	1.401	**22.45**	9.86	**92.98**
~~GP~~	1.15	4.74	3.18	15.50	1.752	1.78	1.84	316.56
proposed	**17.07**	**54.11**	**46.72**	82.36	**1.364**	22.18	**10.04**	93.03

Table 5. Ablation study on the object terms. ↑ means larger numbers are better, ↓ means smaller numbers are better. ~~FM~~ represents a setting of the proposed framework with the feature matching loss term removed, while ~~GP~~ represents a setting of the proposed framework without the gradient penalty loss. The default setting renders better performances on various criteria most of the time. We use the `FUNIT-1` model for this experiment.

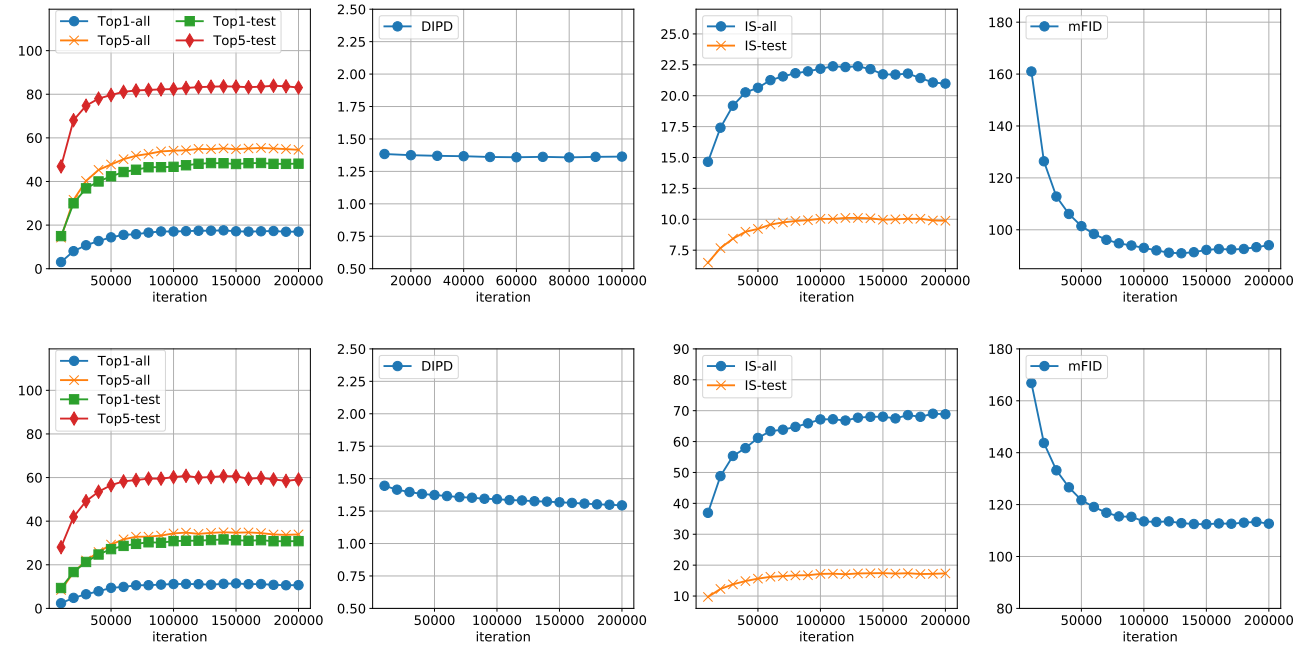


Figure 8. Few-shot image translation performance vs. training iterations. Top row: results on the Animal Faces dataset; bottom row: results on the North American Birds dataset.

Method	# of generated Samples	Split 1	2	3	4	5	Average Accuracy
Baseline	0	38.81 ± 0.01	41.99 ± 0.03	39.13 ± 0.01	37.05 ± 0.02	36.82 ± 0.01	38.76
FUNIT	10	41.20 ± 0.41	46.25 ± 0.27	42.65 ± 0.41	40.75 ± 0.20	39.39 ± 0.31	42.05
	50	41.24 ± 0.16	46.27 ± 0.07	43.15 ± 0.06	41.01 ± 0.19	39.43 ± 0.09	42.22
	100	41.01 ± 0.18	46.72 ± 0.05	42.89 ± 0.09	40.73 ± 0.20	39.33 ± 0.04	42.14
S&H [15]	10	39.87 ± 0.47	42.69 ± 0.34	41.42 ± 0.39	39.95 ± 0.58	38.64 ± 0.42	40.51
	50	39.93 ± 0.15	42.62 ± 0.28	40.89 ± 0.09	39.31 ± 0.17	38.44 ± 0.13	40.24
	100	40.05 ± 0.31	41.72 ± 0.19	41.29 ± 0.16	41.33 ± 0.21	39.39 ± 0.16	40.76

Table 6. One-shot accuracies on the 5 splits of the Animal Faces dataset when using generated images and 1 real image. The average accuracy over 5 independent runs is reported per split (different set of generated images is sampled each time).

Method	# of generated Samples	Split 1	2	3	4	5	Average Accuracy
Baseline	0	30.71 ± 0.02	29.04 ± 0.01	31.93 ± 0.01	29.59 ± 0.01	30.64 ± 0.02	30.38
FUNIT	10	32.94 ± 0.49	33.29 ± 0.25	35.15 ± 0.22	31.20 ± 0.20	34.48 ± 0.58	33.41
	50	32.92 ± 0.34	33.78 ± 0.25	35.04 ± 0.10	31.80 ± 0.14	34.66 ± 0.17	33.64
	100	33.83 ± 0.16	33.99 ± 0.12	36.05 ± 0.14	32.01 ± 0.10	36.09 ± 0.19	34.39
S&H [15]	10	30.55 ± 0.11	31.96 ± 0.30	34.18 ± 0.19	30.65 ± 0.09	31.49 ± 0.24	31.77
	50	31.39 ± 0.07	30.59 ± 0.11	33.60 ± 0.05	30.92 ± 0.20	31.81 ± 0.18	31.66
	100	30.83 ± 0.10	32.03 ± 0.09	34.39 ± 0.17	31.12 ± 0.10	32.23 ± 0.15	32.12

Table 7. One-shot accuracies on the 5 splits of the North Amercian Birds dataset when using generated images and 1 real image. The average accuracy over 5 independent runs is reported per split (different set of generated images is sampled each time).

Figure 9. FUNIT−1 versus AdaIN style transfer [18] for few-shot image translation.

Method	# of generated samples	Average Accuracy: Animal Faces	N. A. Birds
Prototypical Net. [39]	0	34.51	28.49
Prototypical Net. + FUNIT	10	40.07	33.63
	50	39.85	33.74
	100	40.47	34.28

Table 8. 1-shot accuracies on the Animal Faces and the North American Birds datasets upon using generated images with the prototypical networks method [39], averaged over 5 splits.

G. Latent Space Interpolation

We explore the latent space learned by the class encoder. In Figure 11, we use t-SNE to visualize the class code in a two dimensional space. It can be seen that images from similar classes are grouped together in the class embedding space.

Figure 10. Failure cases. The typical failure cases of the proposed FUNIT model include generating hybrid objects (*e.g.* column 1, 2, 3, and 4), ignoring input content images (*e.g.* column 5 and 6), and ignoring input class images (*e.g.* column 7).

Figure 12 shows interpolation results by keeping the content code fixed while interpolating the class code between those of two source class images. Interestingly, we find that by interpolating between two source classes (Siamese cat and Tiger) we can sometimes generate a target class (Tabby cat) that the model has never observed. This suggests that the class encoder learns a general class-specific representation, thus enabling generalization to novel classes.

H. Few-Shot Classification

As mentioned in the main paper, we conduct an experiment using images generated by the FUNIT generator to

Figure 11. 2-D representation of the class code using t-SNE for 5000 images across 50 source classes. Please zoom-in for details.

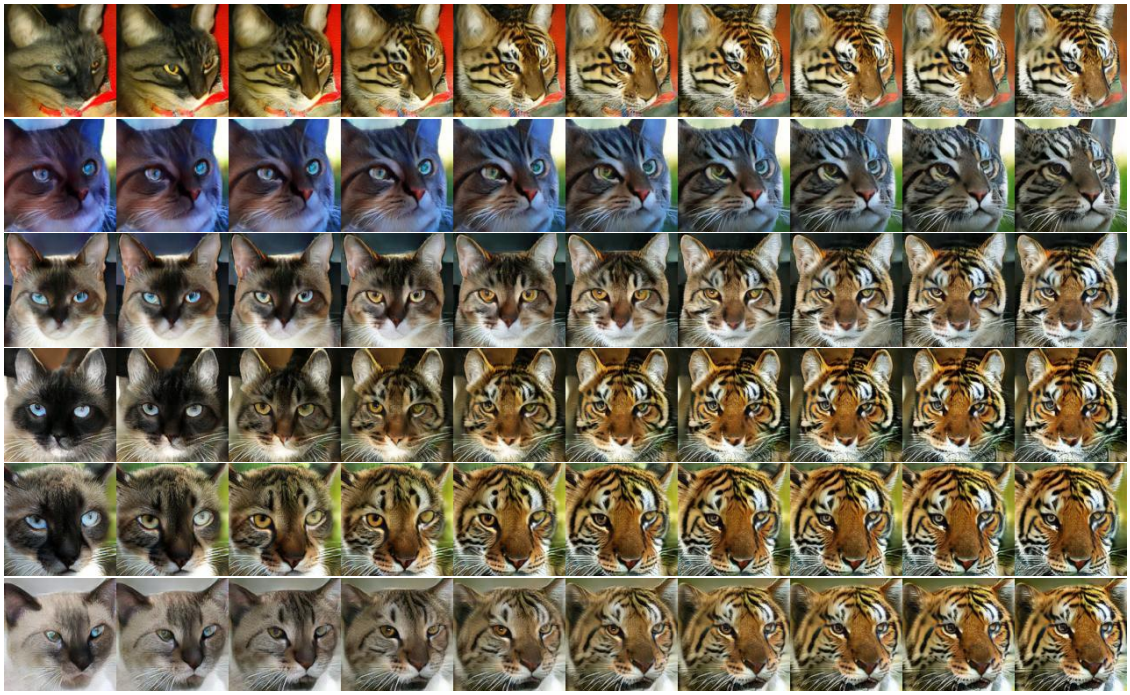


Figure 12. Interpolation by keeping the content code fixed while interpolating between two class codes of source classes.

	Setting	Top1-all ↑	Top5-all ↑	Top1-test ↑	Top5-test ↑	DIPD ↓	IS-all ↑	IS-test ↑	mFID ↓
Animal	CycleGAN-Unfair-All	46.02	71.51	63.44	89.60	1.417	19.91	12.77	90.34
	UNIT-Unfair-All	41.93	67.19	60.76	88.42	**1.388**	20.18	12.29	92.22
	MUNIT-Unfair-All	66.25	86.59	78.45	95.10	1.670	21.99	16.68	71.18
	StarGAN-Unfair-All	39.72	66.18	61.24	89.52	1.392	13.29	9.79	141.57
	FUNIT-Unfair-All	**83.00**	**89.12**	**91.08**	**99.88**	1.445	**25.92**	**23.45**	**32.73**

Table 9. Comparison to the competing methods when using all the images for training.

train classifiers for novel classes in the one-shot setting, using the Animal Faces and North American Birds datasets. Following the setup in Hariharan *et al*. [15], we create 5 different one-shot training splits where each has a training, validation, and test set. The training set consists of $|\mathbb{T}|$ images, one image from each of the $|\mathbb{T}|$ test classes. The validation set consists of 20-100 images from each test classes. The test set consists of remaining test class images.

We use the FUNIT generator to generate a synthetic training set by using the images in the classification training set as the class image input and randomly sampled images from the source classes as the content image input. We train a classifier using both of the original and synthetic training sets. We compare our method against the Shrink and Hallucinate (S&H) method of Hariharan *et al*. [15], which learns to generate final layer features corresponding to novel classes. We use a pretrained 10-layer ResNet network as the feature extractor, which is pretrained purely using the source class images, and train a linear classifier over target classes. We find it crucial to weight the loss on generated images lower than that on real images. We conduct an exhaustive grid search on the weight value as well as the weight decay value using the validation set and report the performance on the test set. For a fair comparison, we also perform the same exhaustive search for the S&H method.

In Table 3 of the main paper, we report performance of our method and the S&H method [15] over different number of generated samples (*i.e*., images for FUNIT and features for the S&H) on two challenging fine-grained classification tasks. Both methods perform better than the baseline classifier that uses just the single provided real image per novel class. Using our generated images, we obtain around 2% improvement over the S&H method that generates features.

The base 10-layer ResNet network is trained for 90 epochs, with an initial learning rate of 0.1 decayed by a factor of 10 every 30 epochs. Weight decay for the linear classifier over novel classes is chosen from 15 logarithmically spaced values between and including 0.000001 and 0.1. The loss multiplier for loss on generated images and features is chosen from 7 logarithmically spaced values between and including 0.001 and 1. The values for weight decay and loss multiplier are chosen based on the best validation set accuracy obtained while training on Split #1. These values are then fixed and used for all remaining splits 2-5. The task of learning an L2 regularized classifier using fixed features is a convex optimization problem, and we use line search with the L-BFGS algorithm, and thus do not have to specify a learning rate.

In Tables 6 and 7, we report test accuracies and their associated variances on one-shot learning for all 5 one-shot splits of the Animal Faces and the North American Birds datasets. In all experiments, we only learn a new classifier layer using features extracted from a network trained on the set of classes used to train the image generator.

Our method can also be used in conjunction with existing few-shot classification approaches. In Table 8, we show 1-shot classification results obtained using the Prototypical Networks method [43], which assigns to a test sample the label of the closest prototype (cluster center) obtained from the given few train samples. Clearly, using our generated samples together with the 1 provided sample per class at test time to compute class prototype representations helps improve accuracy on both datasets considered by over 5.5%.

I. More Translation Results

In Figure 13, 14, 15, and 16, we show additional few-shot translation results for the animal face image translation task, the bird image translation task, the flower image translation task, the food image translation task, and the face image translation task. For the face translation, each class is defined by a person identify. The experiment is conducted using the celebrity face dataset [31]. All the results are computed using FUNIT-5.

J. FUNIT-All

To evaluate performance of the proposed method in the standard image-to-image translation setting where target classes images are available in the training time, we trained a FUNIT model using all the training images in the animal face dataset. The resulting model is called FUNIT-Unfair-All. In this setting, there is no unseen classes in the test time. As shown in Table 9, our FUNIT-Unfair-All outperforms state-of-the-art image-to-image translation models including CycleGAN [55], UNIT [29], MUNIT [19], and StarGAN [8]. (For models that can only handle translation between two domains. We trained multiple of them for evaluation.) This shows that the proposed FUNIT model is also a competitive multi-class image-to-image translation model.

Figure 13. Additional visualization results on the few-shot animal face image translation task. All the results are computed using the same `FUNIT-5` model. The model can be re-purposed for generating images of a dynamically specified target class in the test time by having access to 5 images from the target class. The variable $\boldsymbol{x}$ is the input content image, $\boldsymbol{y}_1$ and $\boldsymbol{y}_1$ are 2 out of the 5 input target class images, and $\bar{\boldsymbol{x}}$ is the translation output. We find that the animal face in the translation output has a similar pose to the input content image but the appearance is akin to the appearance of the animal faces in the class images.

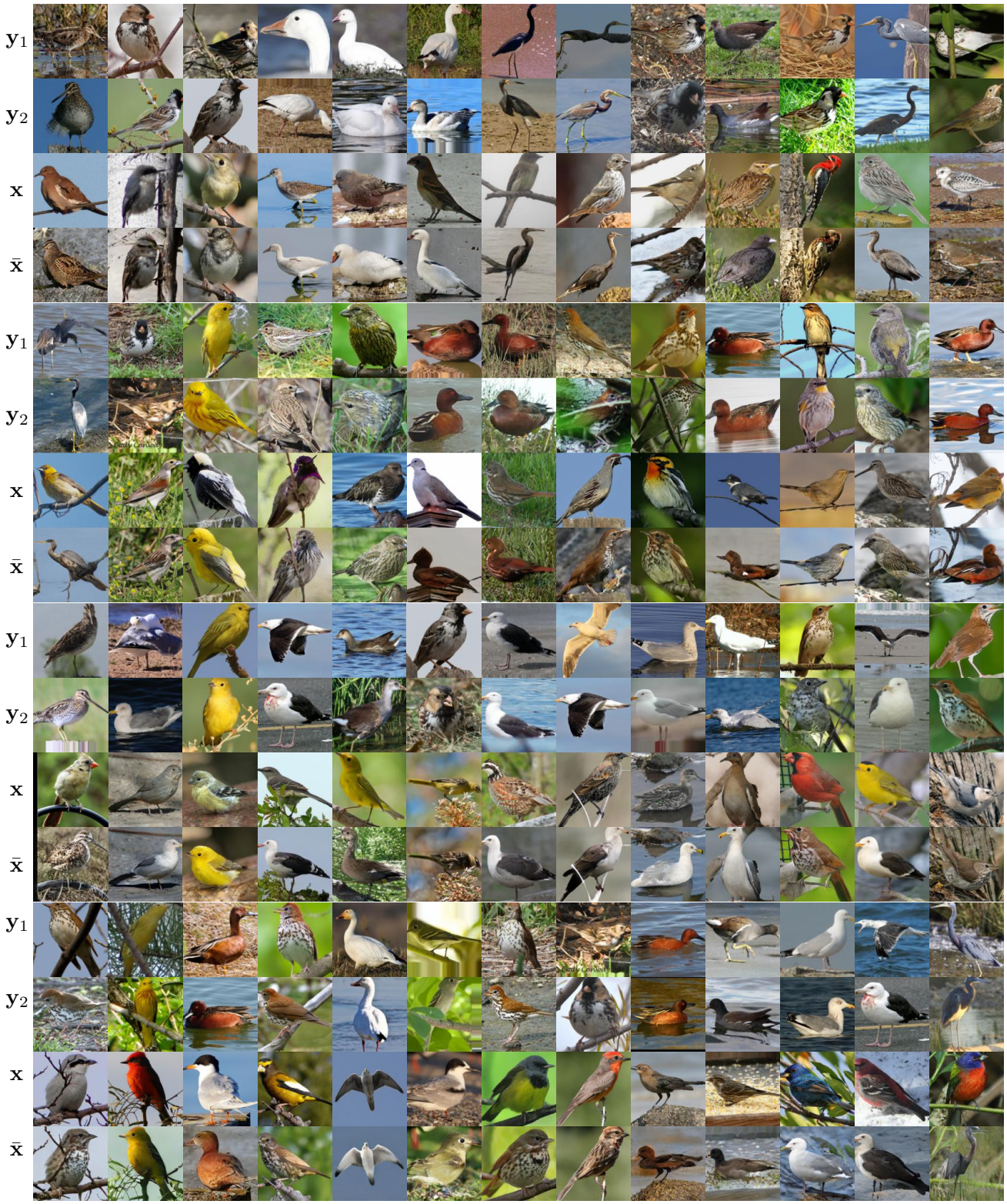


Figure 14. Additional visualization results on the few-shot bird image translation task. All the results are computed using the same `FUNIT-5` model. The model can be re-purposed for generating images of a dynamically specified target class in the test time by having access to 5 images from the target class. The variable $\boldsymbol{x}$ is the input content image, $\boldsymbol{y}_1$ and $\boldsymbol{y}_1$ are 2 out of the 5 input target class images, and $\bar{\boldsymbol{x}}$ is the translation output. We find that the bird in the translation output has a similar pose to the input content image but the appearance is akin to the appearance of the birds in the class images.

Figure 15. Additional visualization results on the few-shot flower and food image translation tasks. All the results for the same task are computed using the same `FUNIT-5` model. The model can be re-purposed for generating images of a dynamically specified target class in the test time by having access to 5 images from the target class. The variable $\boldsymbol{x}$ is the input content image, $\boldsymbol{y}_1$ and $\boldsymbol{y}_1$ are 2 out of the 5 input target class images, and $\bar{\boldsymbol{x}}$ is the translation output. For flower translation, we find the flowers in the output and input image have a similar pose. For food translation, the bowl and plate remain at the same location while the food are changed from one kind to the other.

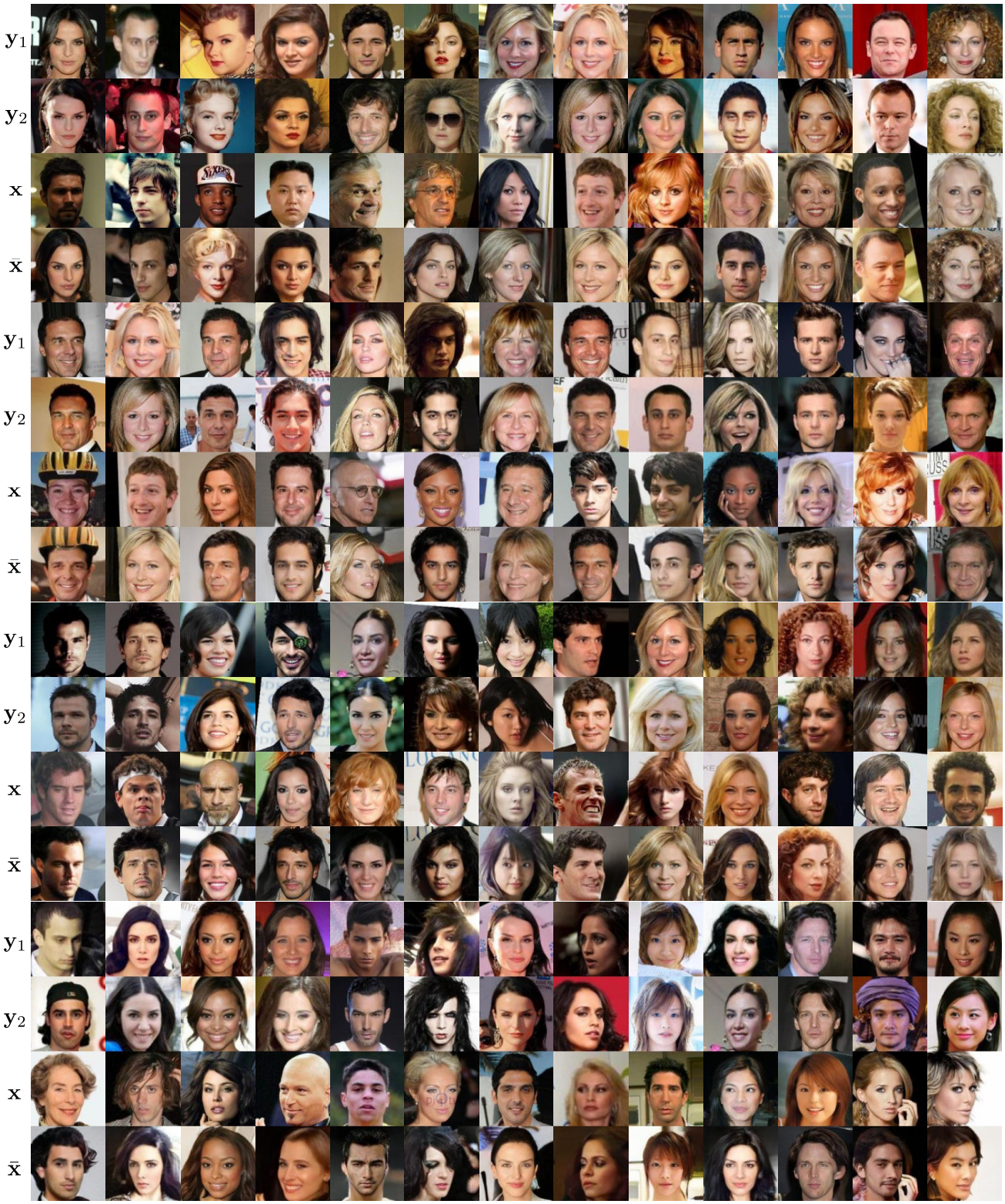


Figure 16. Visualization results on the few-shot face image translation task. All the results are computed using the same `FUNIT-5` model. The model can be re-purposed for generating images of a dynamically specified target class in the test time by having access to 5 images from the target class. The variable $\boldsymbol{x}$ is the input content image, $\boldsymbol{y}_1$ and $\boldsymbol{y}_1$ are 2 out of the 5 input target class images, and $\bar{\boldsymbol{x}}$ is the translation output. We find that the face in the translation output has a similar pose to the input content image but the appearance is akin to the appearance of the faces of the target person.

Few-Shot Unsupervised Image2Image Translation Paper Summary

Prepared By: Sushant Gautam
Follow me on LinkedIn
https://github.com/sushant097/30-Days-GANs-Paper-Reading

This paper try to overcome one of the main problem of GAN that needs huge amount of training samples to generate synthetic images, where given a few target class samples, this architecture translate content to target image which uses adversarial mechanism to train.

This model achieves this few-shot generation capability by coupling an adversarial training scheme with a novel network design.

They proposed FUNIT framework aims at mapping an image of a source class to an analogous image of an unseen target class by leveraging a few target class images that are made available at test time. Example: If only 5 target samples are available then, it is called as FUNIT-5.

Only couple of target images are sufficient for training GAN that learn in an unsupervised way to translate source image to target image.

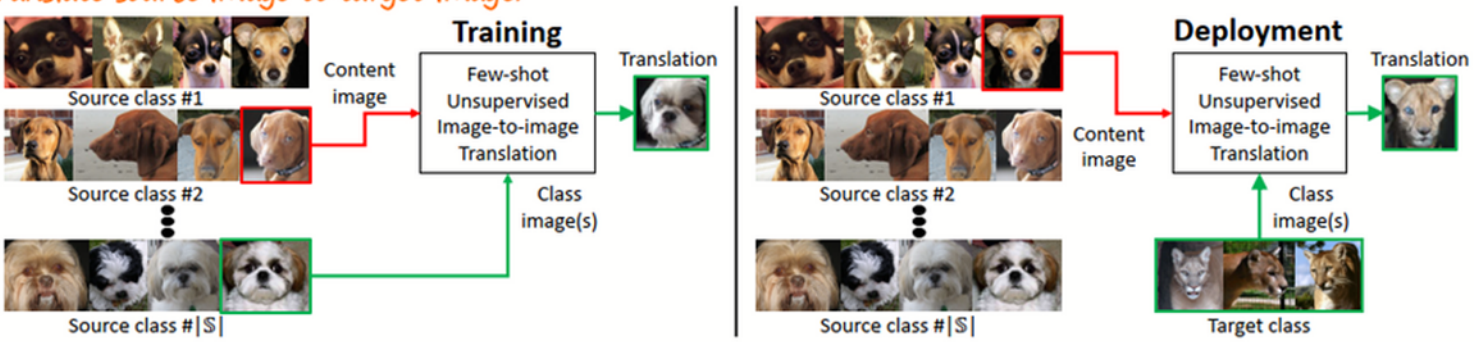


Figure 1. **Training.** The training set consists of images of various object classes (source classes). We train a model to translate images between these source object classes. **Deployment.** We show our trained model very few images of the target class, which is sufficient to translate images of source classes to analogous images of the target class even though the model has never seen a single image from the target class during training. Note that the `FUNIT` generator takes two inputs: 1) a content image and 2) a set of target class images. It aims to generate a translation of the input image that resembles images of the target class.

FUNIT framework works as follows:

1. During training they use set of source classes (various animal species) and a few target images.
2. They use adversarial mechanism to train Generator that leverage the few target images to translate any source class image to analogous images of the target class

Also, if we give the same model few images from a different object class (target), it has to translate any source class images to analogous images of the different novel object class.

One important difference in compare to conditional image generators in existing unsupervised setting is, here Generator G simultaneously takes a content image x and a set of K class images as input and produce output image x' (translated image).

Architecture:

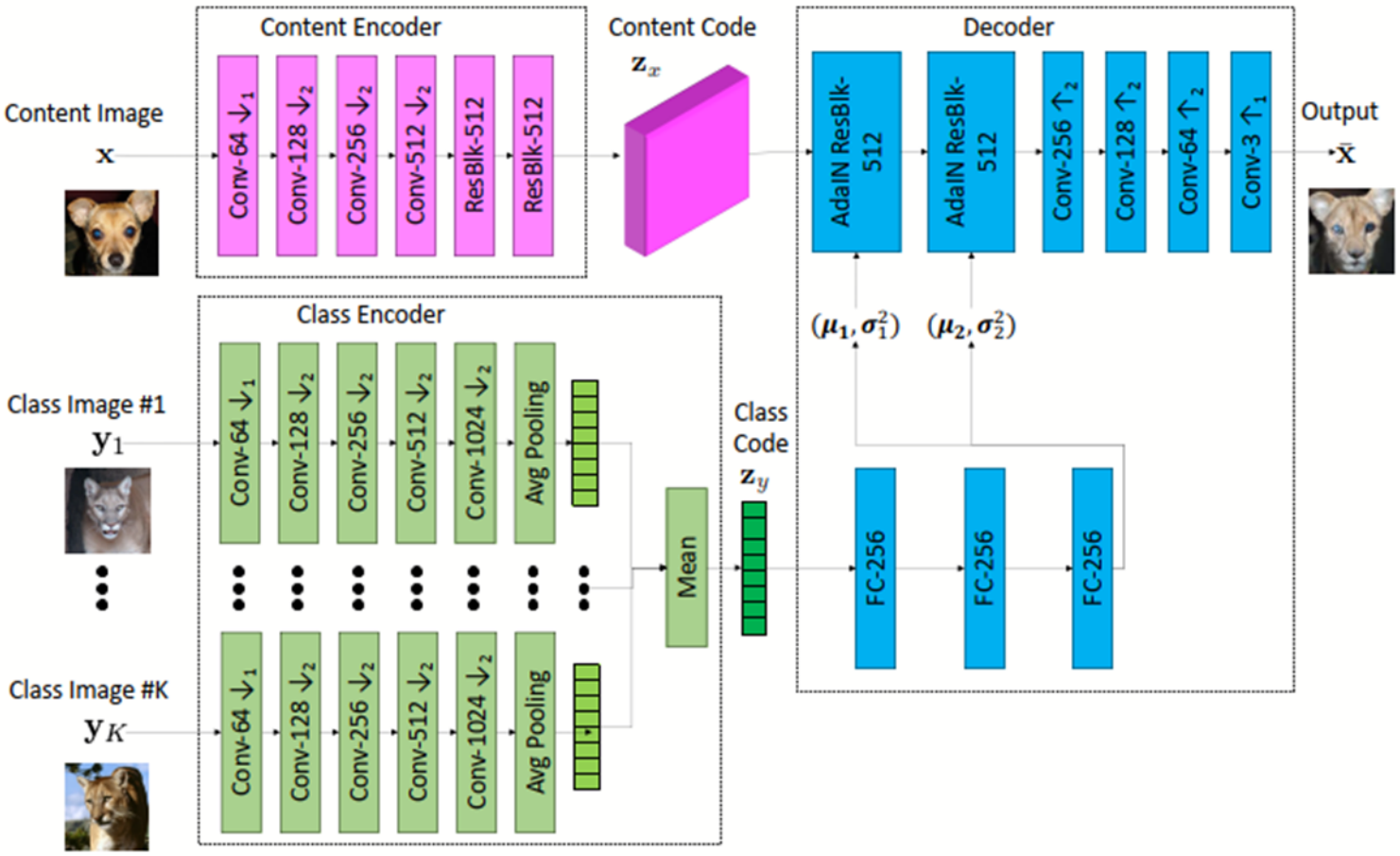


Figure 6. Visualization of the generator architecture. To generate a translation output $\bar{x}$, the translator combines the class latent code z_y extracted from the class images $y_1, ...y_K$ with the content latent code z_x extracted from the input content image. Note that nonlinearity and normalization operations are not included in the visualization.

Generator includes Content Encoder, Class Encoder and Decoder. It first maps each input class image to an intermediate latent code using a VGG-like network. These latent vectors are then element-wise averaged to produce the final class latent code.

Content Encoder which encodes content image and output content code Zx. It includes Convolution with downsampling and ResBlk layer.

Class Encoder which encodes target class image, for each target image is passed through Class Encoder and finally mean is taken which produce class code Zy.

Decoder takes content code Zx, and pass through Convolutional upsampling layers. Additionally, class code Zy is used as mean and variance for Adaptive Instance Normalization (AdaIn). Finally, translated output x' image is generated which is analogous to target class images.

Discriminator: It is Patch GAN discriminator. So, it can judge the finer details in generated image and have fewer parameters which is crucial for learning with few samples.

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-||𝕊||`

where $||\mathbb{S}||$ is the number of source classes.

Loss Function:

Train the proposed FUNIT framework by solving a mnimax optimization problem given by:

$$\min_D \max_G \mathcal{L}_{\text{GAN}}(D, G) + \lambda_{\text{R}} \mathcal{L}_{\text{R}}(G) + \lambda_{\text{F}} \mathcal{L}_{\text{FM}}(G) \quad (3)$$

where $\mathcal{L}_{\text{GAN}}$, $\mathcal{L}_{\text{R}}$, and $\mathcal{L}_{\text{F}}$ are the GAN loss, the content image reconstruction loss, and the feature matching loss. The GAN loss is a conditional one given by

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, D) =& E_{\mathbf{x}}\left[-\log D^{c_x}(\mathbf{x})\right] + \\ & E_{\mathbf{x},\{\mathbf{y}_1,\ldots,\mathbf{y}_K\}}[\log\left(1 - D^{c_y}\left(\bar{\mathbf{x}}\right)\right)] \quad (4)\end{aligned}$$

We assume the content image belongs to object class c_x while each of the K class images belong to object class c_y. In general, K is a small number and c_x is different from c_y. We will refer G as the few-shot image translator.

The content reconstruction loss helps G learn a translation model. Specifically, when using the same image for both the input content image and the input class image (in this case K = 1), the loss encourages G to generate an output image identical to the input.

$$\mathcal{L}_{\text{R}}(G) = E_{\mathbf{x}}\left[||\mathbf{x} - G(\mathbf{x}, \{\mathbf{x}\})||_1^1\right]. \quad (5)$$

The feature matching loss regularizes the training. We first construct a feature extractor, referred to as Df , by removing the last (prediction) layer from D . We then use Df to extract features from the translation output $\bar{x}$ and the class images {y1 , …, yK} and minimize

$$\mathcal{L}_{\text{F}}(G) = E_{\mathbf{x},\{\mathbf{y}_1,\ldots,\mathbf{y}_K\}}\left[||D_f(\bar{\mathbf{x}})) - \sum_k \frac{D_f(\mathbf{y}_k)}{K}||_1^1\right]. \quad (6)$$

Experiments:

They set λR = 0.1 and λF = 1 . Use RMSProp optimizer with learning rate 1e-4. They use hinge version of GAN loss and the real gradient penalty regularization.

They also train the FUNIT model using K = 1 since author desire it to work well even when only one target class image is available at test time. In the experiments, author evaluate its performance under K = 1, 5, 10, 15, 20 . Each training batch consists of 64 content images, which are evenly distributed on 8 V100 GPUs in an NVIDIA DGX1 machine.

They use Animal faces, birds, flowers and foods datasets to perform experiments.

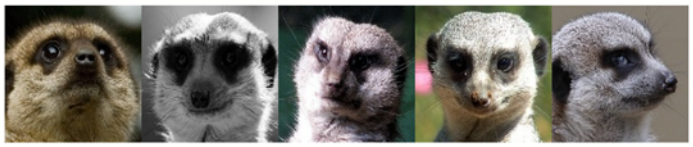
Output:

Figure 2. Visualization of the *few-shot* unsupervised image-to-image translation results. The results are computed using the FUNIT-5 model. From top to bottom, we have the results from the animal face, bird, flower, and food datasets. We train one model for each dataset. For each example, we visualize 2 out of 5 randomly sampled class images $\mathbf{y}_1\mathbf{y}_2$, the input content image $\mathbf{x}$, and the translation output $\bar{\mathbf{x}}$. The results show that FUNIT generate plausible translation outputs under the difficult few-shot setting where the models see no images from any of the target classes during training. We note that the objects in the output images have similar poses to the inputs.

Content Image

to an output meerkat image

by using a set of 5 example meerkat images



Compare to StarGAN BaseLine Model:

Figure 3. Visual comparison of few-shot image-to-image translation performance. From left to right, the columns are input content images $\mathbf{x}$, the two input target class images $\mathbf{y}_1$ $\mathbf{y}_2$, translation results from the unfair `StarGAN` baseline, translation results from the fair `StarGAN` baseline, and results from our framework.

Limitations:

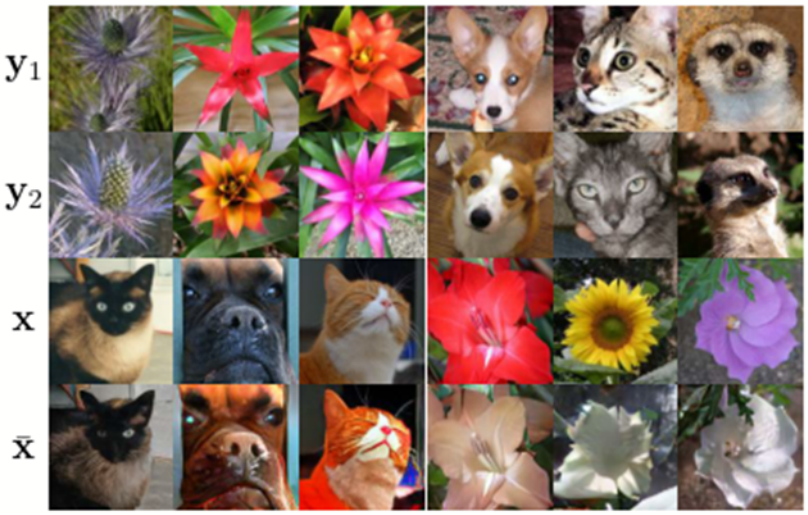


Figure 5. Limitations of the proposed framework. When the appearance of a unseen object class is dramatically different to the appearances of the source classes, (*e.g.* flower and animal face). The proposed `FUNIT` framework fails to generate meaningful translation outputs.