ABSTRACT
Few-shot image generation and few-shot image translation are two related tasks, both of which aim to generate new images for an unseen category with only a few images. In this work, we make the first attempt to adapt few-shot image translation method to few-shot image generation task. Few-shot image translation disentangles an image into style vector and content map. An unseen style vector can be combined with different seen content maps to produce different images. However, it needs to store seen images to provide content maps and the unseen style vector may be incompatible with seen content maps. To adapt it to few-shot image generation task, we learn a compact dictionary of local content vectors via quantizing continuous content maps into discrete content maps instead of storing seen images. Furthermore, we model the autoregressive distribution of discrete content map conditioned on style vector, which can alleviate the incompatibility between content map and style vector. Qualitative and quantitative results on three real datasets demonstrate that our model can produce images of higher diversity and fidelity for unseen categories than previous methods.

KEYWORDS
Few-shot Learning; Image Generation; Quantization

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

1 INTRODUCTION
In recent years, deep generative models [5, 19, 20, 32] can generate high-quality images but require amounts of training data. However, some long-tail categories or newly emerging categories only have limited data [16]. There is a risk of overfitting to train or finetune generative models on a small amount of data [1, 2]. Therefore, it is imperative to consider training on seen categories with sufficient training images and adapting to an unseen category with only a few images. This task is referred to as few-shot image generation [6, 14–16], which transfers knowledge from seen categories to generate diverse and realistic images for unseen categories.

In the remainder of this paper, the images from seen (resp., unseen) categories are dubbed as seen (resp., unseen) images for brevity. The existing few-shot image generation methods can be roughly classified into optimization-based methods, fusion-based methods, and transformation-based methods. The optimization-based approaches FIGR [6] and DAWSON [25] finetuned the unconditional generative adversarial network (GAN) for each unseen category with meta-learning algorithms, but the generated unseen images are unrealistic. In contrast, fusion-based and transformation-based methods are built upon conditional GAN, which takes one or several conditional images from the same category and generates new images for this category. They reduce the burden of finetuning and support instant adaptation to unseen categories [2]. For fusion-based methods, GMN [2], MatchingGAN [14], and F2GAN [16] fused multiple conditional images with matching procedure to produce new images. However, the generated images are similar to conditional images. For transformation-based methods, DAGAN
[1] and DeltaGAN [15] combined random vectors with conditional image to produce new images. However, DAGAN [1] failed to produce diverse images while DeltaGAN [15] may generate undesired unseen images due to the mismatch between random vector and conditional image.

As a closely related task to few-shot image generation, few-shot image translation [27, 35] focuses on translating a seen image to an unseen image. Previous few-shot image translation works like FUNIT [27] disentangle the latent representation into category-invariant content map (e.g., pose) and category-specific style vector (e.g., appearance). In the testing stage, given an unseen image, its style vector can be combined with the content maps of seen images to produce more unseen images for this unseen category, which is equivalent to translating seen images to this unseen category.

In this work, we make the first attempt to adapt representative few-shot image translation method FUNIT to few-shot image generation by solving two critical issues. The first issue is that FUNIT relies on the seen images to provide content maps. It is resource-consuming to store a large number of seen images or content maps of seen images. The number of content maps is also limited by the number of seen images. Inspired by VQVAE [41] which explores discrete representations for compression, we divide the content maps of seen images into local content vectors, in which each local content vector represents the content information of each local part in the image, and compress continuous local content vectors into discrete local content vectors via vector quantization (see Section 5.3). Then, we model the autoregressive distribution of local content vectors, so that we can sample local content vectors autoregressively to form a significant amount of content maps. The second issue of FUNIT is that the randomly selected content maps for an unseen image may be incompatible with the style vector, resulting in poor translation results as shown in Figure 1. To solve the incompatibility problem, we aim to find compatible content maps for a given unseen image. Specifically, we extend the abovementioned autoregressive distribution to conditional autoregressive distribution, which is autoregressive distribution of local content vectors conditioned on a style code. In this way, given an unseen image, we can sample local content vectors autoregressively conditioned on its style vector, and the obtained content maps are expected to be compatible with the style vector. Since we extend FUNIT with discrete content representation, we name our method Disco-FUNIT.

Our contributions can be summarized as follows: 1) We make the first attempt to adapt few-shot image translation to few-shot image generation, which bridges the gap between two research fields; 2) To avoid the reliance on seen images in the testing stage, we explore discrete local content vectors under the disentanglement framework; 3) To solve the incompatibility issue between style vector and content map, we propose to model conditional autoregressive distribution; 4) Our method can produce diverse and realistic images based on a single image, exceeding existing few-shot image generation methods.

2 RELATED WORK

Few-shot Image Generation: As the earliest few-shot image generation methods, Bayesian-based methods [22, 33] were proposed to generate new images for simple concepts in few-shot setting. Recently, deep learning based few-shot image generation methods can be roughly classified into optimization-based methods [6, 25], fusion-based methods [2, 14, 16], and transformation-based methods [1, 15]. Our method can be categorized as the transformation-based method, because we can transform an unseen image to more unseen images from the same category.

Note that some more recent works [24, 29, 34, 45] focused on adapting the generative model pretrained on a large dataset to a small dataset with a few examples, which are also called few-shot image generation. However, their setting is quite different from ours. Firstly, these methods adapt from one source domain to another target domain, while our method adapts from multiple seen categories to unseen categories. Secondly, they need to finetune the model for each unseen domain, which is very tedious. Instead, we can make instant adaptation to unseen categories without finetuning.

Few-shot Image Translation: In [3], one image of unseen category is regarded as an exemplar to guide image translation. FUNIT [27] utilized a disentanglement framework to combine the content maps of seen images with the style vectors of unseen images to produce translated images belonging to unseen categories. Built upon FUNIT, COCO-FUNIT [35] proposed a content-conditioned style encoder architecture to adjust the style code. SEMIT [46] proposed a semi-supervised method for few-shot image translation by reducing the amount of required labeled data during training. In this work, we build upon FUNIT and adapt it to few-shot image generation.

Vector Quantization: The feature vectors learned from convolutional network always contain a lot of redundant information [10, 17, 44]. To compress these learned features, vector quantization methods [41] have been proposed to construct a dictionary of discrete vectors to approximate the actual continuous vectors. In other words, one original continuous vector can be represented by its nearest center in the dictionary. Vector quantization can significantly reduce the size of the storage footprint. In this work, we utilize the vector quantization technique to create a dictionary of local content vectors, which serves as a foundation of modeling the compatibility between content map and style vector.

Autoregressive Models: The autoregressive models are a type of generative models that have been widely used for image generation, in which the autoregressive distribution can be modeled by different sequence-to-sequence models like PixelCNN, LSTM, and Transformer. PixelCNN [30] modeled the distribution of a natural image using the elementary chain rule over pixels. LSTM [38] exploited the hierarchical relationships between complete images and image portions. Transformer modeled the long-range interactions between local image representations [31, 47]. In this work, we leverage Transformer to model the autoregressive distribution.

3 BACKGROUND ON FUNIT

Since our goal is adapting few-shot image translation method FUNIT [27] to few-shot image generation, we first have a brief introduction to FUNIT, which is basically a disentanglement network to
Figure 2: Our method consists of two training stages: disentanglement with discrete content map and style-conditioned content map autoregression. In the first stage, we jointly learn an FUNIT model and a dictionary \( \mathcal{D} \), during which continuous content map \( \hat{C}^x \) is quantized into discrete content map \( \hat{C}^x \). In the second stage, we learn a content map generator \( T_{\text{cls}} \) to model the autoregressive distribution of \( \hat{C}^x \) conditioned on the style vector \( s^x \). At test time, given an unseen image \( z \), \( T_{\text{cls}} \) samples diverse content maps \( \{\hat{C}^x\} \) autoregressively conditioned on the style vector \( s^x \) to produce diverse images \( \{\hat{z}\} \).

Disentangle content map and style vector. All categories are split into non-overlapping seen categories and unseen categories. FUNIT is trained on seen categories. Then, the learned model can leverage an image of an unseen category to translate any seen image to this unseen category.

FUNIT consists of a generator \( G \) and a multi-class discriminator \( D \), where \( G \) is constructed with a content encoder \( E_c \), a style encoder \( E_s \), and a decoder \( F \). During training, we randomly sample an image \( x \) from a seen category \( F^x \) and an image \( y \) from another seen category \( l^y \). We use \( E_c \) to extract content map \( C^x = E_c(x) \), where \( w \times h \) (resp., \( d_c \)) is the spatial size (resp., channel dimension) of content map. We use \( E_s \) to extract style vector \( s^y = E_s(y) \), where \( d_s \) is the dimension of style vector. Then, content map \( C^x \) and style vector \( s^y \) are fed into the decoder \( F \) to produce translated image \( \hat{y} \):

\[
\hat{y} = F(C^x, s^y).
\]

(1)

\( \hat{y} \) integrates the content information of \( x \) and the style information of \( y \). Because style is category-specific information, \( \hat{y} \) belongs to the same category as \( y \). The multi-class discriminator \( D \) is designed to ensure the realism of translated image \( \hat{y} \) given the category label \( l^y \) by optimizing adversarial loss:

\[
\begin{align*}
\mathcal{L}_D &= \mathbb{E}_y [\max(0, 1 - D^y(\hat{y})) + \mathbb{E}_x [\max(0, 1 - D^y(y))],
\mathcal{L}_{GD} &= -\mathbb{E}_y [D^y(\hat{y})],
\end{align*}
\]

(2)

where \( D^y \) (resp., \( D^x \)) denotes the discriminator corresponding to the category \( l^y \) (resp., \( l^x \)).

When applying \( E_c \) and \( E_s \) to the same image \( x \), \( F \) is encouraged to reconstruct the input image \( x \) with a reconstruction loss \( \mathcal{L}_R = \mathbb{E}_x ||x - \hat{x}|| \), where \( \hat{x} = E_s(x) \).

To stabilize the adversarial training, feature matching loss \( \mathcal{L}_{FM} = \mathbb{E}_y,\hat{y} ||D(y) - D(\hat{y})||_1 \) is designed to minimize the distance between \( D(y) \) and \( D(\hat{y}) \), where \( D \) is feature extractor constructed by removing the last layer from \( D \). The overall objective of FUNIT is summarized as follows,

\[
\mathcal{L}_{\text{funit}} = \mathcal{L}_D + \mathcal{L}_{GD} + \lambda_R \mathcal{L}_R + \lambda_F \mathcal{L}_{FM},
\]

(3)

in which \( \lambda_R \) and \( \lambda_F \) are hyper-parameters. After training, an input image is disentangled into content map and style vector. In the inference stage, given an image \( z \) from an unseen category \( F^z \), the style vector of \( z \) can be combined with the content map of any seen image, yielding a new image with category label \( F^l \).

4 OUR METHOD

In this section, we introduce the process of adapting FUNIT to few-shot image generation. As shown in Figure 2, our few-shot image generation method consists of two training stages. The first stage is disentanglement with discrete content map and the second stage is style-conditioned content map autoregression. We refer to the image providing content (resp., style) as content (resp., style) image. In the first stage, given images \( x \) and \( y \) from two seen categories, we train an FUNIT model as in Section 3, but the content map \( C^x \) is quantized into discrete content map \( \hat{C}^x \) via vector quantization method [41]. In the second stage, conditioned on the style vector \( s^x \) of \( x \), we model the compatibility between style vector \( s^x \) and
discrete content map \( \hat{C}^z \) using a content map generator \( T_{c,z} \). At test time, given an image \( z \) from unseen category \( F \), our content map generator \( T_{c,z} \) can sample local content vectors autoregressively to form diverse content maps \( \hat{C}^z \) which are compatible with the style vector \( s^2 \). Then, different content maps \( \{ \hat{C}^z \} \) can be combined with \( s^2 \) to produce diverse and realistic images \( \{ \tilde{z} \} \) from unseen category \( F \).

4.1 Disentanglement with Discrete Content Map

Based on FUNIT, given an image \( x \), content encoder \( E_c \) extracts its content map \( C^x \in \mathbb{R}^{w \times h \times d_c} \). To avoid the heavy storage cost of content maps and remove the redundant information from content maps, we divide a content map to local content vectors and quantize them into discrete local content vectors.

Specifically, we spatially split the content map into local content vectors, with each local content vector representing the content information of a local part (e.g., eye, nose) in the original image. Similar to [21], content map \( C^x \) is ordered from left to right and from top to bottom to form a sequence \( \{ e_i^{x|N} \} \), in which \( N = w \times h \) and each \( e_i^x \) is a \( d_c \)-dim local content vector.

To quantize continuous local content vectors into discrete ones, we learn a dictionary of local content vectors \( D = \{ e_k \}_{k=1}^K \) where \( K \) is the number of items in the dictionary. Then, continuous local content vectors \( \{ e_i^{x|N} \} \) are quantized into discrete local content vectors \( \{ \hat{e}_i^{x|N} \} \) by looking up its nearest neighbour in the dictionary \( D \). Specifically, each local content vector \( e_i^x \) can be replaced with its nearest item \( e_k \) in the dictionary \( D \) by comparing the distances between \( e_i^x \) and all items in \( D \). This process can be represented by

\[
e_i^x = e_k, \quad \text{where} \quad k = \arg \min_j \| e_i^x - e_j \|_2.
\]

Then, the sequence of discrete local content vectors \( \{ \hat{e}_i^{x|N} \} \) can be reorganized into a discrete content map \( \hat{C}^x \in \mathbb{R}^{w \times h \times d_c} \) according to the order from left to right and from top to bottom. As in Section 3, translation and reconstruction is performed based on \( \hat{C}^x \). On the one hand, \( \hat{C}^x \) and \( s^2 \) are used to produce the translated image \( \tilde{y} = F(\hat{C}^x, s^2) \). On the other hand, \( \hat{C}^x \) and \( s^2 \) are used to reconstruct \( x \). Following [4], we adopt a straight-through gradient estimator. This estimator simply copies the gradients from the decoder \( F \) to the content encoder \( E_c \) and achieves back-propagation through the non-differentiable quantization procedure, which allows the disentanglement model and the dictionary of local content vector \( D \) to be trained in an end-to-end manner with the following loss function:

\[
L_{\text{vq}}(G, D) = \lambda_{\text{vq}} L_{\text{vq}} + L_{\text{fun}},
\]

in which \( L_{\text{vq}} = \| \text{sg} (C^x) - \hat{C}^x \|_2^2 + \| \text{sg} (\hat{C}^x) - C^x \|_2^2 \cdot \text{sg} 1 \) represents the stop-gradient operation and \( \| \text{sg} (C^x) - \hat{C}^x \|_2^2 \) denotes commitment loss [41]. Another loss term \( L_{\text{fun}} \) is identical with Eqn. 3 and \( \lambda_{\text{vq}} \) is a hyper-parameter.

4.2 Style-conditioned Content Map Autoregression

After the first training stage, we still require a large amount of images to provide adequate discrete content maps \( \hat{C} \). To support stochastic sampling to produce diverse and plausible discrete content maps, we model the autoregressive distribution of discrete content maps of real images.

In particular, the process of generating a discrete content map can be formulated as a sequence-to-sequence prediction. Thus, we can learn a probabilistic model \( T_c \) to predict the distribution of next local content vector based on the known local content vectors. Given the historic sequence \( \hat{C}_{<i} = \{ \hat{e}_j^{x|N} \} (\hat{C}_{<1} = 0) \), the probabilistic model predicts the distribution of \( \hat{e}_i \), that is, \( p(\hat{e}_i|\hat{C}_{<i}) \). In detail, each local content vector \( e_i^x \) can be encoded into one-hot vector according to its index \( k \) in dictionary \( D \). The probabilistic model takes in the sequence \( \hat{C}_{<i} \) and predicts the probability of the index \( k \) of \( \hat{e}_i \). The probabilistic model can be realized by many sequence-to-sequence models like PixelCNN [50], LSTM [48], and Transformers [8, 31, 49]. We opt for Transformer due to its ability of capturing long-range dependency and superior performance in various tasks [9, 18]. The technical details of Transformer are left to Supplementary due to space limitation. Overall, the probabilistic model \( T_c \) tends to maximize the likelihood of discrete content maps of real images \( x \):

\[
p(\hat{C}^x) = \prod_{i=1}^N p(\hat{e}_i^x | \hat{C}^x_{<i}).
\]

After training the probabilistic model, we can sample \( \{ \hat{e}_i^{x|N} \} \) according to the predicted probability \( p(\hat{e}_i|\hat{C}_{<i}) \) in the sequential order to construct a plausible discrete content map. However, the sampled discrete content map may be incompatible with the style vector and the same issue also exists for previous few-shot image translation methods like FUNIT (see Section 5.3). Therefore, we extend \( T_c \) to a style-conditioned probabilistic model \( T_{c,s} \) to generate discrete content map conditioned on a given style vector. The only difference between \( T_c \) and \( T_{c,s} \) is that we append a style vector to the input sequence. Specifically, \( T_{c,s} \) takes \( s^2 \) and the historic sequence \( \hat{C}_{<i} \) to predict the probability of \( \hat{e}_i \). Overall, the probabilistic model \( T_{c,s} \) tends to maximize the conditional likelihood of discrete content maps of real images \( x \):

\[
p(\hat{C}^x|s^2) = \prod_{i=1}^N p(\hat{e}_i^x | s^2, \hat{C}^x_{<i}).
\]

The loss function of training \( T_{c,s} \) can be written as

\[
L_T = E_X[-\log p(\hat{C}^x|s^2)].
\]

During testing, given an image \( x \) from an unseen category \( F \), we extract its style vector \( s^2 \) from style encoder \( E_s \). The probabilistic model \( T_{c,s} \) functions as a content map generator, which generates amounts of discrete content maps \( \{ \hat{C}^z \} \) conditioned on \( s^2 \). These discrete content maps \( \{ \hat{C}^z \} \) and \( s^2 \) are fed into the decoder \( F \) to produce new images \( \{ \tilde{z} \} \) belonging to category \( F \).

5 EXPERIMENTS

5.1 Experimental Setup

Datasets We conduct experiment on Flowers [28], Animal Faces [7], and NABirds [42] datasets. Following the split setting of FUNIT [27], a total of 102 (resp., 149, and 555) categories of Flowers (resp., Animal Faces, and NABirds) dataset are split into 85 (resp., 119, and 444) seen categories and 17 (resp., 30, and 111) unseen categories.

Implementation Following FUNIT [27], we set \( \lambda_R = 0.1, \lambda_F = 1 \) without further tuning. We set \( \lambda_{\text{vq}} = 0.8, K = 1024 \), and \( N = 16 \times 16 \).
after a few trials by observing the quality of generated images in the training stage. We use Pytorch 1.7.0 to implement our model, which is distributed on RTX 2080 Ti GPU. For the first (resp., second) stage, we employ Adam optimizer with learning rate of \(1e^{-4}\) (resp., \(5e^{-5}\)), and the batch size is set to 8 (resp., 4). Following [27], image resolution in our experiments is \(128 \times 128\).

**Baselines** We compare our method Disco-FUNIT with few-shot image translation methods including FUNIT [27] and COCO-FUNIT [35], as well as few-shot image generation methods with same setting as ours including DAWSON [25], MatchingGAN [14], F2GAN [16], Dagan [1], and DeltaGAN [15]. Note that few-shot image translation methods rely on seen images at test time. Besides, we can use our method without the second training stage as a quantized few-shot image translation method, which is referred to as “Disco-FUNIT (1st stage)”.

### 5.2 Quantitative Evaluation

We perform quantitative evaluation on the realism and diversity of the generated images, and also evaluate the category-preserving property using the downstream few-shot classification task.

**Evaluation of Realism and Diversity** To evaluate the realism and diversity of images generated by different methods, Fréchet Inception Distance (FID) [13] and Learned Perceptual Image Patch Similarity (LPIPS) [52] are calculated on three datasets. We use FID to measure the distance between generated unseen images and real unseen images. In particular, we use ImageNet-pretrained Inception-V3 [39] model as the feature extractor. Then, FID is calculated between the extracted features of generated unseen images and those of real unseen images. The LPIPS is used to assess the diversity of generated unseen images. The average of pairwise distances among generated images is computed for each unseen category, and the final LPIPS score is obtained by averaging over all unseen categories.

Considering that the number of conditional images in fusion-based methods MatchingGAN and F2GAN is configurable, we use 3 conditional images following F2GAN [16]. If \(H\) images are provided for each unseen category at test time, we refer to this configuration as the \(H\)-shot setting. We report the 3-shot results for all methods, and also report 1-shot results for Dagan, DeltaGAN, FUNIT, COCO-FUNIT, and our method. Following [15], we employ each method to generate 128 images for each unseen category for calculating FID and LPIPS either in 1-shot setting or 3-shot setting. We also try using more images to calculate FID, but observe no significant difference.

For Dagan, DeltaGAN, FUNIT, and COCO-FUNIT, and our method in 3-shot setting, we generate 128 images by randomly sampling one conditional/style image each time. We summarize the results in Table 1. In both 3-shot and 1-shot settings, we can see that our method has the lowest FID and the highest LPIPS, indicating that it can generate more diverse and realistic images than baseline methods. Disco-FUNIT (1st stage) has already outperformed FUNIT and COCO-FUNIT, which demonstrates that discrete content maps can help remove noisy and redundant information from continuous content maps for better disentanglement. The second stage in our method can further improve the performance (Disco-FUNIT v.s. Disco-FUNIT (1st stage)), which shows that our style-conditioned content map generator can sample diverse content maps compatible with the given style vector for producing new images of higher fidelity. Another observation is that few-shot image translation methods (i.e., FUNIT and COCO-FUNIT) generally outperform few-shot image generation baselines by using seen images to provide content information during testing.

**Few-shot Classification** To demonstrate that our generated images belong to the desired category and can benefit few-shot classification, we conduct experiments in \(L\)-way \(M\)-shot setting following [37, 43], in which the average accuracy over several evaluation episodes is calculated. In each evaluation episode, \(L\) unseen categories are chosen randomly and \(M\) images are chosen randomly from each selected unseen category. The remaining images from \(L\) unseen categories form the test set, while the selected \(L \times M\) images form the training set. We use the seen images to pretrain ResNet18 [12] and remove the last FC layer to extract features.
Table 2: Accuracy(%) of different methods on three datasets in few-shot classification setting. Note that fusion-based methods MatchingGAN [14] and F2GAN [16] are not usable in 1-shot setting.

| Method                  | Flowers 10-way 1-shot | Flowers 10-way 5-shot | Animal Faces 10-way 1-shot | Animal Faces 10-way 5-shot | NABirds 10-way 1-shot | NABirds 10-way 5-shot |
|-------------------------|------------------------|------------------------|-----------------------------|-----------------------------|-----------------------|-----------------------|
| MatchingNets            | 40.96                  | 56.12                  | 36.54                       | 50.12                       | 33.59                 | 46.08                 |
| MAML                    | 42.95                  | 58.01                  | 35.98                       | 49.89                       | 34.12                 | 46.21                 |
| RelationNets            | 48.18                  | 61.03                  | 45.32                       | 58.12                       | 40.59                 | 49.68                 |
| MTL [36]                | 54.34                  | 73.24                  | 52.54                       | 70.91                       | 44.21                 | 59.64                 |
| DN4 [23]                | 56.76                  | 73.96                  | 53.26                       | 71.34                       | 43.53                 | 60.51                 |
| MatchingNet-LFT [40]    | 58.41                  | 74.32                  | 56.83                       | 71.62                       | 46.16                 | 60.67                 |
| DPGN [50]               | 58.95                  | 74.56                  | 57.18                       | 72.02                       | 46.39                 | 58.38                 |
| DeepEMD [51]            | 59.12                  | 73.97                  | 58.01                       | 72.71                       | 47.68                 | 58.29                 |
| GCNET [26]              | 57.61                  | 72.47                  | 56.64                       | 71.53                       | 46.35                 | 60.36                 |
| MatchingGAN [14]        | -                      | 74.09                  | -                           | 70.89                       | -                     | 57.16                 |
| F2GAN [16]              | -                      | 75.02                  | -                           | 73.19                       | -                     | 59.78                 |
| DeltaGAN [15]           | 61.23                  | 77.09                  | 60.31                       | 74.59                       | 49.05                 | 61.58                 |
| FUNIT [27]              | 55.98                  | 73.12                  | 56.61                       | 69.12                       | 46.08                 | 56.84                 |
| COCO-FUNIT [35]         | 57.18                  | 74.91                  | 58.01                       | 72.78                       | 47.58                 | 59.04                 |
| Disco-FUNIT (1st stage) | 62.59                  | 77.52                  | 60.96                       | 75.11                       | 50.34                 | 61.98                 |
| Disco-FUNIT             | 63.11                  | 78.89                  | 61.85                       | 75.71                       | 51.84                 | 63.18                 |

Figure 3: Images generated by F2GAN [16], DeltaGAN [15], and our Disco-FUNIT in 3-shot setting on two datasets (from top to bottom: Animal Faces and NABirds). The conditional images are in the left three columns.

for unseen images. In L-way M-shot setting, following [16], our method generates 512 new images to augment each of L categories in each evaluation episode. We extract features of L × (M + 512) unseen images in the training set and train a linear classifier, which is then applied to the test set.

We compare our method with state-of-the-art few-shot classification methods, including the representative methods MatchingNets [43], RelationNets [37], MAML [11] as well as the state-of-the-art methods MTL [36], DN4 [23], MatchingNet-LFT [40], DPGN [50], DeepEMD [51], and GCNET [26]. In each evaluation episode, no augmented images are added to the training set for these baselines. Instead, images from seen categories are used to train those few-shot classifiers while adhering to their original training strategy. MAML [11] and MTL [36] models must be fine-tuned in each evaluation episode based on the training set. We also compare our method with few-shot image generation methods MatchingGAN, F2GAN, DeltaGAN as well as few-shot image translation methods FUNIT, COCO-FUNIT. We adopt the same augmentation strategy as our method in each evaluation episode. We show the averaged accuracy across 10 evaluation episodes on three datasets in Table 2 using 10-way 1-shot/5-shot as examples. It can be seen that our...
method produces the best results on all datasets, demonstrating the high quality of images generated by our method. Another observation is that few-shot image translation methods (i.e., FUNIT and COCO-FUNIT) underperform some few-shot image generation methods (e.g., DeltaGAN, our full method), because the content map may also contain style information and disturb the category of generated images [15].

5.3 Qualitative Evaluation

Comparison with Few-shot Image Generation Baselines We exhibit some example images generated by F2GAN, DeltaGAN, and our method conditioned on 3 unseen images in Figure 3. For DeltaGAN and our method, the generated images are arranged according to the conditional images (two generated images for each conditional image). We can observe that the images generated by F2GAN are still similar to one of the conditional images, while DeltaGAN may generate distorted images with blurry details. In contrast, our method can generate images of higher diversity and fidelity, because our discrete content map can help remove redundant and noisy information and style-conditioned content map generator can sample content maps compatible with style vectors of the conditional images. In Figure 4, we also show some example images generated by our method on three datasets in 1-shot setting. More comparison images are visualized in Supplementary.

In Figure 4, we also show some example images generated by our method on three datasets in 1-shot setting. In detail, given an unseen image, conditioned on its style vector, we can sample 9 different style-aware discrete content maps from learned transformer, which combined with style code is fed into decoder to generate 9 images. We can see that the generated images can keep the style of the unseen image and have desired content information (such as the pose of animals and birds, and sketch of flowers). More results can be found in Supplementary.

Comparison with Few-shot Image Translation Baselines We visualize some example images generated by FUNIT, COCO-FUNIT, Disco-FUNIT (1st stage), and our full method in Figure 5. For FUNIT, COCO-FUNIT, Disco-FUNIT (1st stage), the images are generated based on the style image in the first row and the content image in the second row. For our full method, we sample content maps without using the content image in the second row. We can observe that the images generated by FUNIT contain many artifacts, whereas COCO-FUNIT can generate higher quality images, but translated images sometimes lose details and look far from style image (see column 3-4 and 7-8). In contrast, images generated by Disco-FUNIT (1st stage) are more realistic and maintain the details of unseen style images (e.g., ear in column 1, 7, 8, eye in column 3-4, and texture in column 4-8), since discrete content maps can remove redundant and noisy information from continuous content maps to achieve better disentanglement. However, the above methods may generate poor images (see column 4 and 6), when the style vector is incompatible with the content map (e.g., the style image and content image have too divergent poses). In contrast, our full method can sample compatible content maps to produce images with appearance details similar to the style image and plausible poses, which demonstrates the effectiveness of our style-conditioned content map generator. More generated images from Disco-FUNIT (1st stage) (resp., Disco-FUNIT) are visualized in Supplementary.

5.4 Ablation Studies

By taking Animal Faces dataset as an example, we study the effect of each stage in our method and alternative network designs. For each ablated method, we report FID, LPIPS, and the accuracy of few-shot classification (10-way 1-shot) as in Section 5.2 in Table 3. The analyses of hyper-parameters (e.g., content map size, dictionary
length, and transformer layers) are left to Supplementary. due to space limitation.

**Effect of Each Stage in Our Method** First, we remove the second training stage from our method, which is referred to as "Disco-FUNIT (1st stage)" in Table 3. The difference between FUNIT and "Disco-FUNIT (1st stage)" is that content map is continuous in FUNIT but discrete in "Disco-FUNIT (1st stage)". Based on Table 3 and Figure 5, we can observe that discrete content map can substantially improve the quality of translated images, because content map quantisation can cooperate with FUNIT to remove redundant information from content map for better disentanglement.

To study the effect of style-conditioned content map autoregression stage, we compare "Disco-FUNIT (1st stage)" with our full method in Table 3, which shows that our content map generator $T_{c|s}$ can further improve the quality of generated images by sampling content maps compatible with the style vector. Besides, our content map generator $T_{c|s}$ only relies on unseen images and learned dictionary at test time, which adapts "Disco-FUNIT (1st stage)" to few-shot image generation. We compare storage consumption of seen images and learned dictionary in Supplementary.

**Alternative Designs of Content Map Generator** To evaluate the effectiveness of the style-conditioned content map generator $T_{c|s}$, we explore different ways to model the autoregressive distribution of discrete content map. The first one is "$p(\hat{C}^x)$" as in Eqn. 6. The second one is replacing conditional style vector $s^x$ with the style vector from a different image $s^y$, which is referred to as "$p(\hat{C}^x|s^y)$". From Table 3, we can see that "$p(\hat{C}^x)$" is worse than Disco-FUNIT, which demonstrates the advantage of sampling content map conditioned on the style vector. We also observe that "$p(\hat{C}^x|s^y)$" is even worse than "Disco-FUNIT (1st stage)", which proves that it is necessary to model the conditional likelihood based on paired contents map and style vector from real images.

6 INVESTIGATION OF STYLE-CONDITIONED CONTENT MAP AUTORegression

To further investigate the effectiveness of our style-conditioned content map generator $T_{c|s}$ in the second stage, we show some example images generated by Disco-FUNIT (1st stage), one variant of Disco-FUNIT ("$p(\hat{C}^x)$"), and Disco-FUNIT. As introduced in Section 5.4, $p(\hat{C}^x)$ models the autoregressive distribution of content maps without the conditional style vector.

For Disco-FUNIT (1st stage), the images are generated based on the unseen style image in the first row and the seen content image in the second row. In detail, given a style image, we randomly select 50 seen content images to produce 50 translated images. Then, we calculate the loss value of each translated image according to Eqn 8, in which the smaller loss value indicates that the discrete content map and the style vector are more compatible. After sorting these 50 images by loss value in an increasing order, we show six images with the smallest loss values and four images with the largest loss values for "$p(\hat{C}^x)$" (resp., Disco-FUNIT) in row 4 (resp., 5) in Figure 6.

By comparing Disco-FUNIT (1st stage) and Disco-FUNIT, it can be seen that our content map generator can sample diverse content maps to produce more satisfactory images with high probability, while Disco-FUNIT (1st stage) may produce undesired images due to incompatible content maps of seen content images (see column 5-10 in row 3). By comparing "$p(\hat{C}^x)$" and Disco-FUNIT, we can see that the images generated by "$p(\hat{C}^x)$" are diverse but occasionally distorted, e.g., partial images with small loss values (column 5-6 in row 4) and all images with large loss values (column 7-10 in row 4). The observation is consistent with higher FID and higher LPIPS reported in Table 3, which also indicates that the generated images with rich diversity may be unrealistic due to distortion or artifacts. In contrast, the images generated by Disco-FUNIT with small loss values are rich in details and look more realistic. However, we acknowledge that the images generated by Disco-FUNIT with large loss values may also be distorted (column 9-10 in row 5).

7 CONCLUSION

In this paper, we have proposed to adapt few-shot image translation method to few-shot image generation by learning a compact dictionary of local content vectors and model the compatibility between content map and style vector. Extensive experiments on three real datasets have demonstrated the effectiveness of our proposed Disco-FUNIT.

Acknowledgments

The work is supported by Shanghai Municipal Science and Technology Major Project, China (2021SHZDZX0102), and Shanghai Municipal Science and Technology Key Project (Grant No. 20511100300), and National Science Foundation of China (61902247).
