Few-shot Learning Project

Group 17

December 18, 2020

1 Introduction

Since the birth of deep learning, it has become the mainstream research field of machine learning. However, most tasks require a sufficient number of data to obtain high-performance models. The few-shot image generation algorithm is useful for solving limited data and long-tail distribution in the real world. After training a large number of sample datasets on the seen category and obtaining the available model, we may use this model to generate broad and realistic pictures for unseen category samples, which does not appear in previous datasets. Few-shot image generation seeks to generate more data of a given domain, with a few available training examples. As it is unreasonable to expect to infer the distribution from just a few observations, we seek to leverage a large, related source domain as pretraining. Few-shot image generation can be used for many downstream tasks such as data expansion, small sample classification, etc.

We survey several existing few-shot image generation methods in this project, including F2GAN, DAGAN, FIGR, etc. We apply these methods in different datasets to obtain some generation results. We test all models on the Omniglot, a prevailing few-shot learning dataset, and compare their results.

2 Related work

Few-shot learning. Few-shot learning [12, 22] is first explored in discriminative problem, i.e. few-shot image classification. The main problem is how to improve the generalization performance on the unseen classes while preventing the model from being over-fitted to the few examples. The key is to use prior knowledge, such as data (augment the supervised experience), model (reduce the size of hypothesis space), and algorithm (improve the search in the hypothesis space). For the unseen classes in a few-shot classification, the term *few* refers to a few labels, which means there can be plenty of unlabelled images, and labeled instances are limited. However, in a few-shot image generation, we assume that there are only a few images. No other information about the unseen domain is available. Another big difference with classification is that the generation aims at generating mixed results while the classification only unseen classes to predict a consistent label.

Adapt pre-trained model. Li et al. [15] uses *Elastic Weight Consolidation* to preserve the diversity of the source domain while adapting to the appearance of the target. They adopt a pre-trained model and regularize the changes of the weights.

Meta learning. In meta-learning problems, there is a distribution of tasks, and we would like to obtain an agent that learns fast when presented with a previously unseen task sampled from this distribution. Meta-learning algorithms, likes MAML[4] and Reptile [17] , can be used to improve the generation tasks on unseen data. FIGR[3] is a few-shot image generation method using the Reptile algorithm. Reptile works by repeatedly sampling a task, training on it, and moving the initialization towards the trained weights on that task. It learns a parameter initialization that can be fine-tuned quickly on a new task and uses only first-order derivatives when updating. MAML relies on the direction of the loss function, which does not fit for GAN, so it is not used in FIGR.

Matching networks. Matching networks are memory-assisted networks that leverage an external memory by employing an attention module to learn new concepts quickly. It assumes that the concepts stored are somewhat similar to the new out-of-sample concepts. Bartunov et al. [1] designed generative matching networks conditioning on the additional input dataset that can instantly learn new concepts that were not available in the training data but conform to a similar generative process.

Data Augmentation. Data augmentation [11] enlarges the dataset with new samples. It generates more data by applying transformations to the existing dataset, including random translation, rotations, and flips, and the addition of Gaussian noise. Data Augmentation relies on a prior known invariance.

Generative Adversarial Network. Generative Adversarial Network (GAN) [5] is a generative model based on adversarial learning. It can learn complex joint densities and boast an impressive capacity for realistic generative images. It is widely used in image translation and image generation. Attempting to apply this method in few-shot image learning is a challenging task for the lack of data.

Attention Mechanism Attention module is a sub-module widely used in deep learning algorithms that are mainly used to locate areas of interest.Many attention mechanisms, including spatial attention [23], channel attention [2], and full attention [20]. Two works are mostly related to F2GAN. The method in [14] employs a local attention mechanism to select relevant information from multi-source human images for human image generation, but it fails to capture long-range relevance. Inspired by non-local attention [21][26], F2GAN develop a novel non-local attentional fusion (NAF) module for few-shot image generation.

3 Method

3.1 GAN

GAN includes a generator G and a discriminator D. G maps a random noise vector z to a fake image $\hat{x}$, such that $G(z) = \hat{x}$. D should discriminate fake images $\hat{x}$'s from real images x's, while G is trained to fool D. This adversarial game between the two models leads to G to generate images that resemble real images.

3.2 F2GAN

Using the small sample data $X_s = x_k|_{k=1}^K$, and random interpolation coefficient $a = [a^1, ..., a^K]$ of the same category, F2GAN will generate realistic images of the same type. Non-local Attention Fusion (NAF) module for low-level detail information to generate new pictures x.

For small sample data of the same class $X_s = x_k|_{k=1}^K$ and random interpolation coefficient $a = [a^1, ..., a^K]$, the F2GAN model generates images of the same class. F2GAN uses a random difference coefficient $a = [a^1, ..., a^K]$ and fuses X_s feature which generate from the higher level for features fusion. Then uses the lower-level detail information from Non-local Attentional Fusion (NAF) to generate a new picture x during the resampling process.

F2GAN split categories into two-part, one is seen categories C^s, which is used for F2GAN model training, and C^u is used for image generation. During training, the F2GAN model is used for training the F2GAN model and learn a mapping image, which translates a few conditional images X^S of a seen category. During the testing phase, a few-shot image X_s from unseen category C^u and random interpolation coefficients a are fed into the F2GAN model as it is shown in figure 1.

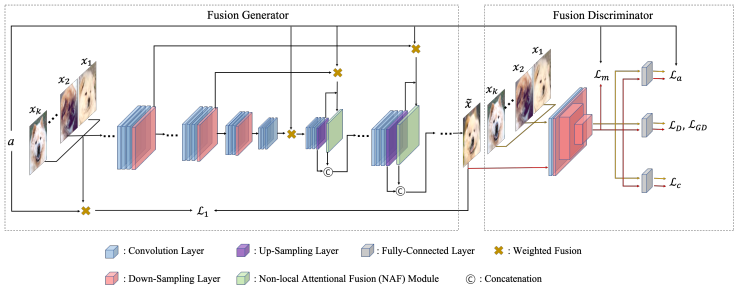


Figure 1: The framework of F2GAN

As shown, F2model has two parts: the generator part and the discriminator part. For generator part, F2GAN use U-net[18] combine ResNet [7]. There are five encoder residual blocks and five decoder residual blocks, and one intermediate block. For $r = 1, 2, 3, \ldots$, the r-th skip connection directs the output from the r-th encoder block to the output from the r-th decoder block, and ξ_r^k to denote the r-th output of decoder block and Φ_r to denote the output feature from the r-th decoder block. The NAF module can be shown as figure 2.

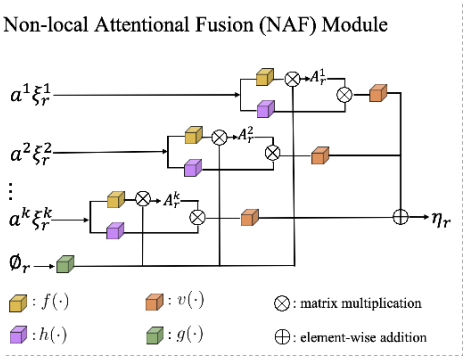


Figure 2: Non-local Attentional Fusion (NAF) module

In NAF module, $f(x)$, $g(x)$, $h(x)$ and $v(x)$ denoted $1 * 1 * \frac{C_r}{8}$ convolutional layer [16]. And weighted reconstruction loss is used to constrain the generated image as Formula 1.

$$L_1 = \sum_{k=1}^{K} a^k ||x_k - x||_1 \tag{1}$$

And the discriminator consists of one convolutional layer followed by five residual blocks. We treat K conditional images $x_k|_k^K = 1$as real images and the generated image $\hat{x}$ fake image. In detail, the average score $D(x)$ for K conditional images and the score $D(\hat{x})$for generated image $\hat{x}$ are calculated for adversarial learning. To stabilize training process, we use hinge adversarial loss . To be exact, the goal of discriminator D is minimizing L_D while the goal of generator is minimizing L_{GD} as Formula 2 and 3.

$$L_D = E_x[max(0, 1) + D(\hat{x})] + E_{x_k}[max(0, 1 - D(x))] \tag{2}$$

$$L_{GD} = -E_{\hat{x}}[D(\hat{x})] \tag{3}$$

Specifically, the last fc layer of the discriminator D is replaced by another fc layer with the output dimension being the number of seen categories. See Formula 4.

$$L_c = -logp(c(x)|x) \tag{4}$$

where $c(x)$ is the ground-truth category of x.

In our fusion generator, when sampling two different interpolation coefficients $a1$ and $a2$, the generated images $G(a1, X_S)$ an $4G(a2, X_S)$ are likely to collapse into the same mode.To guarantee the diversity of generated images, we use two strategies to mitigate mode collapse, one is a variant of mode seeking loss to seek for more modes, the other is establishing bijection between the generated image $\hat{x}$ and its corresponding interpolation coefficient a. The mode seeking loss in was originally used to produce diverse images when using different latent codes. Here,we slightly twist the mode seeking loss to produce diverse images when using different interpolation coefficients. Specifically, we remove the last fc layer of D and use the remaining

feature extractor $\hat{D}$ to extract the features of generated images with different interpolation coefficients. Then, we maximize the ratio of the distance between $\hat{D}(G(a1, X_S))$ and $\hat{D}(G(a2, X_S))$ over the distance between a_1 and a_2, yielding the following mode seeking loss 5:

$$L_m = \frac{||\hat{D}(G(a_1, X_S)) - \hat{D}(G(a_2, X_S))||_1}{||a_1 - a_2||_1} \tag{5}$$

F2GAN apply a fully-connected (fc) layer E to the concatenated feature $[\hat{D}(x_k)$, $\hat{D}(\hat{x})]$, and obtain the similarity scores k between x_k and $\hat{x}$: $s_k = E([\hat{D}(x_k)$, $\hat{D}(\hat{D}(\hat{x}))])$. Then, we apply softmax layer to $s = [s_1, ..., s_K]$ to obtain the predicted interpolation coefficients $\hat{a} = softmax(s)$, which are enforced to match the ground truth a cas Formula 6:

$$L_a = ||\hat{a} - a||_2 \tag{6}$$

So the overall loss function to be minimized is as Formula 7,

$$L = L_D + L_{GD} + \lambda L L_1 + L_c - \lambda_m L_m + \lambda_a L_a \tag{7}$$

in which λ_1,λ_m, and λ_a are trade-off parameters. In the framework of adversarial learning, fusion generator and fusion discriminator are optimized by related loss terms in an alternating manner.

3.3 DAGAN

The architecture of the DAGAN model is shown in Figure 3.

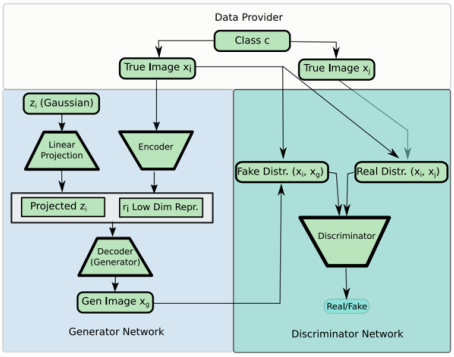


Figure 3: DAGAN Architecture

Input a data $\boldsymbol{x_i}$ to the encoder, we can learn its latent representation $\boldsymbol{r_i} = g(\boldsymbol{x_i})$, which extracts the information needed to generate new data. Then sample the latent Gaussian variables $\boldsymbol{z} = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ from the standard Gaussian distribution, and combine $\boldsymbol{r_i}$ and $\boldsymbol{z}$ with the neural network decoder we can get augmentation data $\boldsymbol{x_g} = f(\boldsymbol{r_i}, \boldsymbol{z})$ that supplements original data $\boldsymbol{x_i}$.

DAGAN uses an improved WGAN [6] critic to train model. It inputs real points $(\boldsymbol{a})$ and generated points (b) to the discriminator:

(a) $\{\boldsymbol{x_i}, \boldsymbol{x_j}\}$: $\boldsymbol{x_j}$ is a second data point which from the same class with input data $\boldsymbol{x_i}$.

(b) $\{\boldsymbol{x_i}, \boldsymbol{x_g}\}$: $\boldsymbol{x_g}$ is the output of the current generator which takes $\boldsymbol{x_i}$ as an input.

The generator is trained to minimize this discriminative capability as measured by the Wasserstein distance. The discriminator network is trained to discriminate between the samples from the real distribution and the fake distribution.

In this model, providing the original input xi to the discriminator is meaningful and vital. By providing information about xi, we can make sure that the generator outputs related but different augmentation data $\boldsymbol{x_g}$. Simultaneously, we do not provide class information, so it has to learn to generalize in consistent ways across all classes. Simultaneously, class information is not provided to the network, which means DAGAN should learn the cross-class transformations. Such a network does not depend on the classes themselves. Therefore it can be applied to unseen novel classes, including few-shot classes.

The DAGAN generator takes the UResNet model (a combination of UNet [18] and ResNet [7]). The UResNet generator has eight blocks, each block having four convolution layers followed by one down-scaling or up-scaling layer. The DAGAN discriminator takes the DenseNet [10] model. The DenseNet was composed of 4 Dense Blocks and 4 Transition Layers.

3.4 FIGR

FIGR [3] uses GAN to generate images. The adapted Reptile pseudo-code for meta-training the model is depicted in Algorithm 1. The algorithm is composed of an outer loop and an inner loop. The inner loop is the K step of the operator U on a copy of the parameters Φ with task τ. Once we have those adapted weight W_τ, we can proceed to the outer loop. We set the gradient of Φ to be equal to $\Phi - W_\tau$. We then take one step with the Adam optimizer. Wasserstein loss is used to make training more stable.

Algorithm 1 FIGR training

Initialize Φ_d, the discriminator parameter vector
Initialize Φ_g, the generator parameter vector
for iteration 1, 2, 3 ... **do**
    Make a copy of Φ_d resulting in W_d
    Make a copy of Φ_g resulting in W_g
    Sample task τ
    Sample n images from X_τ resulting x_τ
    for $K > 1$ iterations **do**
        Generate latent vector z
        Generate fake images y with z and W_g
        Perform step of SGD update on W_d with
         Wasserstein GP loss and x_τ and y
        Generate latent vector z
        Perform step of SGD update on W_g with
         Wasserstein loss and z
    Set Φ_d gradient to be Φ_d - W_d
    Perform step of Adam update on Φ_d
    Set Φ_g gradient to be Φ_g - W_g
    Perform step of Adam update on Φ_g

Once meta-trained, we use a similar process to generate novel images from the sampled class described in Algorithm 2.

Algorithm 2 FIGR generation

Using W_d, a copy of the meta-trained Φ_d
Using W_g, a copy of the meta-trained Φ_g
Sample test task τ
Sample n images as x_τ from X_τ
for $K >= 1$ iterations **do**
    Generate latent vector z
    Generate fake images y with z and W_g
    Perform step of SGD update on W_d with
     Wasserstein GP loss and x_τ and y
    Generate latent vector z
    Perform step of SGD update on W_g with
     Wasserstein loss and z
Generate latent vector z
Generate fake images y

4 Experimental setting

4.1 Datasets

We find several commonly used datasets for few-shot learning after a survey, including MNIST, EMNIST, Omniglot, VGGFace, and FIGR-8. In our experiments, we choose three typical datasets to evaluate the algorithms, which are Omniglot, VGGFace, and FIGR-8.

MNIST MNIST is simple, so it allows us to iterate quickly through model ideas. The MNIST dataset contains 28×28 grayscale images from the ten digits. We use the 60000 training set images for all experiments.

EMNIST The EMNIST dataset is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format and dataset structure that directly matches the MNIST dataset.

Omniglot Omniglot is arguably the *de facto* dataset for few-shot image generation. It contains 1623 unique type of characters originating from 50 alphabets, each of which has been handwritten one time by 20 different individuals. Contrarily to MNIST, Omniglot allows for training our model on a much more considerable amount of classes of images and testing the model's out-of-sample performance in a broader set of classes.

VGGFace The dataset contains 3.31 million images of 9131 subjects (identities), with an average of 362.6 images for each subject. Images are downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity, and profession (e.g., actors, athletes, politicians). The whole dataset is split into a training set (including 8631 identities) and a test set (including 500 identities).

FIGR-8 Clouatre et al. proposes a new dataset, FIGR-8[3], which contain 1548944 images separated in 18409 conceptually different classes. Each class contains at least 8 images, up to a few thousand. The icons are black-and-white representations of objects, concepts, patterns, or designs that have been created by designers and artists and compiled into one data set. Each of those classes containing at least eight images of a similar theme. Every image is of square format $1 \times 200 \times 200$. The dataset is more challenging for meta-learning, as it contains a wide variety of samples inside each class and a substantial amount of classes. Sample taken from the FIGR-8 dataset are pictured in Figure 4.

4.2 Evaluation metrics

The most simple evaluation method is human observation.

We can also train classifier networks to evaluate the result of the generation. All data can be further split into 2 test cases, three validation cases, and a varying number of training cases depending on the experiment. Classifier training is done on the training

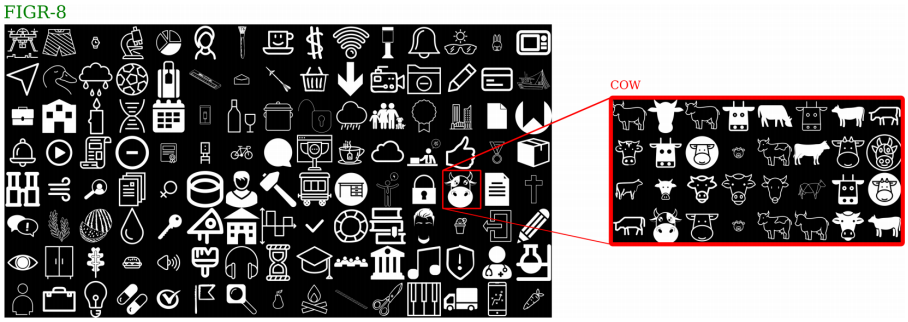


Figure 4: Sample taken from the FIGR-8 dataset

cases for all examples in all domains. Finally, test performance will be reported only on the test cases for the target domain set.

The inception score [25] involves using a pre-trained deep learning neural network model for image classification to classify the generated images. A large number of generated images are classified using the model. Precisely, the probability of the image belonging to each class is predicted. These predictions are then summarized into the inception score. The score seeks to capture two properties of a collection of generated images: Image Quality and Image Diversity.

The Frechet Inception Distance score[9], or FID for short, is a metric that calculates the distance between feature vectors calculated for real and generated images. The FID score is used to evaluate the quality of images generated by generative adversarial networks, and lower scores have been shown to correlate well with higher quality images.

The LPIPS[27] can be used to measure the average feature distance among the generated images. We compute the average pairwise distances among generated images for each category and then compute the average overall unseen categories as the final LPIPS score.

4.3 F2GAN

4.3.1 Settings

Experiments have been done with F2GAN on Omniglot dataset[13]. 1802 (resp. 1200 , 28) categories have been randomly chosen from the total 2395 (resp. 1623,48) categories while training as been seen. Furthermore, 497(resp.,212,10) categories have been randomly choosing as the unseen testing categories.

For experiment, $\lambda_1 = 1$, $\lambda_m = 0.01$, and $\lambda_a = 1$ in Formula 7. The conditional image K has been set to $K = 3$ by balancing the benefit against the cost. There is no need for a larger K for it can only improve a little improvement. With Adam optimizer, the model has trained 200 epochs with a learning rate of 0.0001. F2GAN has been trained in Omniglot for 200 epochs

For batch size B_{size}, we set it to be $B_{size} = 20$ to do a trade-off of better convergence and cost calculation. All experiments were run on TU102 TITAN RTX.

4.3.2 Results

For F2GAN, three metric is used for model Evaluation, which is Inception Scores(IS) [24], Fréchet Inception Distance (FID) [8]. We use the IS metric to evaluate the image quality of the generated images. During the evaluation, the ImageNet pretrained Inception-V3 model[19] has been fine-tuning with unseen categories to calculate the IS for generated images. For the similarities between the generated image and the seen image in the test set, the FID is used to measure it. We remove the last average pooling layer of the ImageNet-pretrained Inception-V3 model as the feature extractor. We compute Fréchet Inception Distance between the generated images and the real images from the unseen categories based on the extracted features.

In the evaluation process, the models trained on the visible categories were used to generate images for each category that the model did not see using the random interpolation coefficient and the conditional pictures of $k = 3$. Enough images can be generated by repeating the above steps. In this way, by adjusting the interpolation coefficient and repeating the above algorithm, we can generate enough generated images.

For each unseen category, 128 images are generated by the algorithm, and then the test generated images are evaluated by the FID mentioned above, is, lips evaluation indicators. For the IS, the higher is better.; the lower the FID, the better the generated images are.

As we can see, figure 5 shows the validation loss curve of F2GAN in the Omniglot dataset. We choose the minimum validation trained model after 170 epoch training as the best F2GAN to generate the figure for an unseen figure.

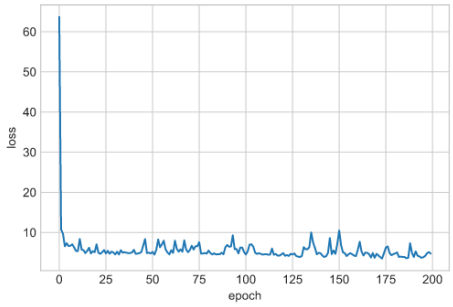


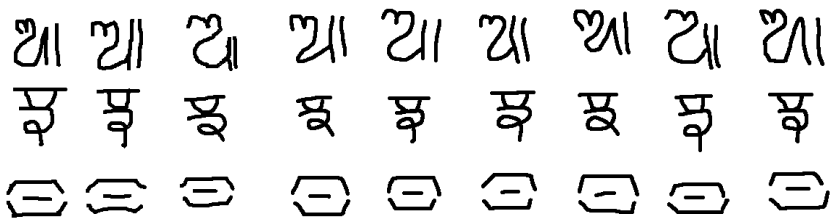
Figure 5: Validation loss curve of F2GAN in the Omniglot experiment

Some examples are shown in Figure 6.

As we can see, the images generated by F2GAN are not closer to inputs with limited diversity. Most figures generate by F2GAN are diverse and realistic because the information of more than one conditional image is fused more coherently in F2GAN.

The evaluation indexes of F2gan are shown in the Table 1.

Figure 6: Images generated by F2GAN (K=3) on Omniglot datasets



(a) Input Figure (b) Generated Figure

Table 1: Quantitative evaluation of images generated by F2GAN

	IS_{mean}	IS_{std}	FID
$Omniglot_{F2GAN}$	1.4257	0.0877	119.2571

4.4 DAGAN

4.4.1 Settings

The DAGAN model we used in the experiment is described in section 3.3. We used a learning rate of 0.0001 and an Adam optimizer with Adam parameters of $\beta_1 = 0$ and $\beta_2 = 0.9$.

The experiment of DAGAN was done on two datasets: Omniglot and VGG-Face.

In the experiment of Omniglot, the Omniglot data was split into the source domain set, validation domain set, and target domain set. The order of the classes was shuffled to make each domain contain diverse samples. Moreover, we defined the first 1200 classes as source domain set, 1201-1600 classes as validation domain set, and 1601-1023 as target domain set. We only used the first 900 classes of source domain set to train our model during the experiment. We trained the DAGAN using the training set and validated the trained DAGAN model in the validation domain. The DAGAN of Omniglot was trained for 300 epochs, and we chose the model with the best validation loss as the final result. Then we test the performance of the best model we found in the target domain by generating some samples.

The experiment of VGG-Face was similar to the Omniglot experiment. The first 1803 of the VGG-Face data was used as the source domain set, and 1804-2300 was the validation domain set. The rest was set as the target domain set. Similarly, the order of the classes was shuffled. We only used the first 600 classes of source domain set to train our model. We trained the VGG-Face DAGAN for 50 epochs and chose the best model in the validation domain set. Then we used the best model to generate some fake images from real images sampled in the target domain and evaluate them.

We used Inception Scores(IS) and Fréchet Inception Distance (FID) as our evaluation metrics. IS and FID are two frequently used indexes to evaluate the quality of images generated by GAN. The IS is positively correlated with the visual quality of generated images, and the FID is designed for measuring similarities between two

sets of images. We randomly sampled ten real images from the target domain set and generated five fake images for each one using spherical interpolation.

4.4.2 Results

The curve of Omniglot validation loss is shown in Figure 7. We chose the trained model in epoch 284, which has the minimum validation loss, as the best DAGAN model to generate unseen images.

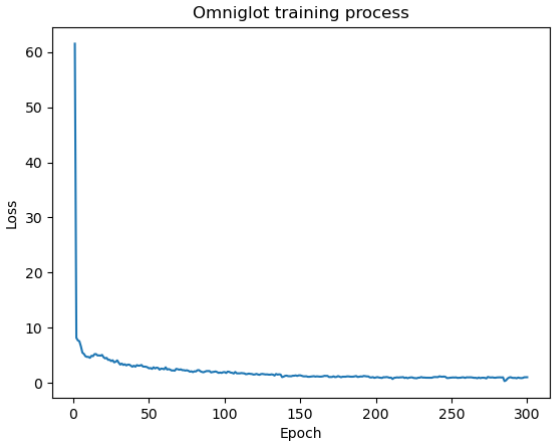


Figure 7: Validation loss curve of DAGAN in the Omniglot experiment

There are two examples generated by DAGAN trained in Omniglot in Figure 8. The image in the top left corner is a real image, and others are generated images.

The IS and FID of Omniglot DAGAN are shown in Table 2.

The curve of VGG-Face validation loss is shown in Figure 9. We chose the trained model in epoch 50, which has the minimum validation loss, as the best DAGAN model to generate unseen images.

There are two examples generated by DAGAN trained in VGG-Face in Figure 10. The image in the top left corner is a real image, and others are generated images.

The IS and FID of VCC-Face DAGAN are shown in Table 2.

Table 2: Quantitative evaluation of images generated by DAGAN

	IS_{mean}	IS_{std}	FID
Omniglot	1.2274739	0.110515326	125.617134
VGG-Face	1.3596112	0.11757518	153.80597

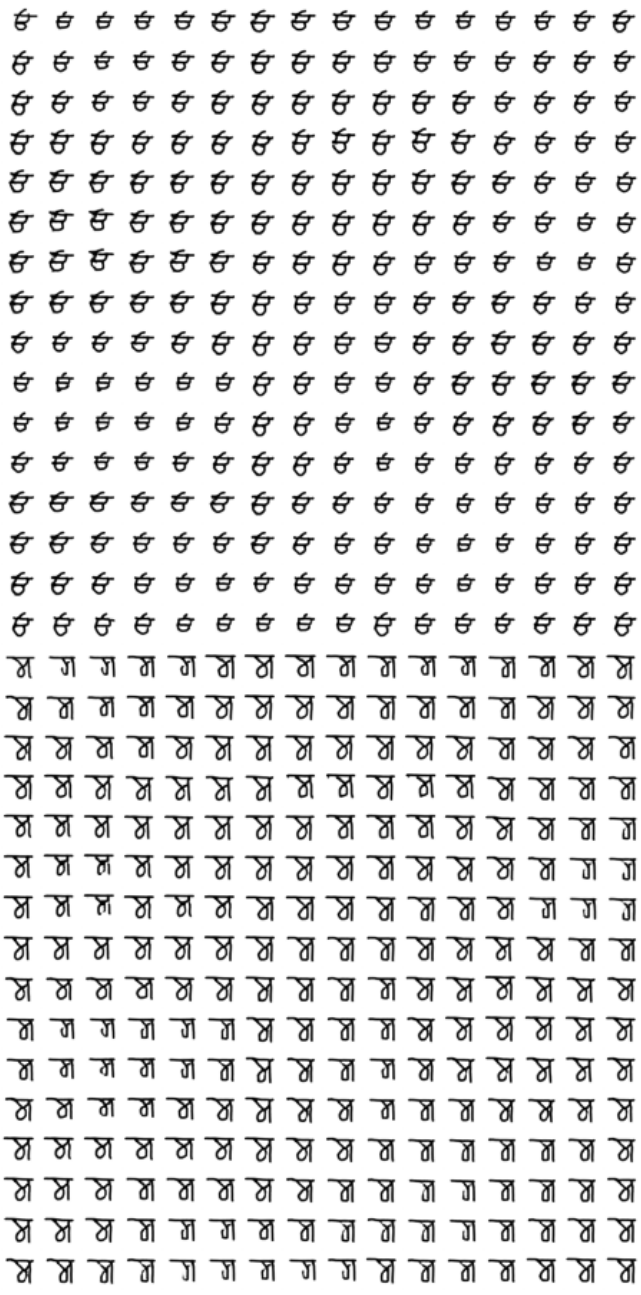


Figure 8: two samples of characters generated by DAGAN

Figure 9: Validation loss curve of DAGAN in the VGG-Face experiment

4.5 FIGR

4.5.1 Settings

Model All models have been trained with Wasserstein loss with gradient-penalty. Both the generator and the discriminator are built with residual neural networks with 18 layers. The discriminator and generator use layer normalization as prescribed in Gulrajani et al. All rectified linear units are Parametric ReLU (PReLU).

Image All images are resized with bilinear interpolation to 32×32. All images are in grayscale format and normalized to have values constrained between -1 and 1. No data augmentation was used.

Training hyperparameters The training batch size $n = 4$. Some class has less than four images, so we allow choosing with repetition. The inner learning rate is 0.0001, and the outer one is 0.00001. Inner loops $K = 10$. Image is resized to 64×64. Grayscale is set to true.

4.5.2 Results

Shown below are the results of generating unseen test classes on our three datasets. The first row of every figure that follows represents the training data. The following three rows are images generated by the model fine-tuned on those data points for ten gradient steps. All present images results on previously unseen test classes.

Omniglot The training classes were all 1623 characters in the dataset minus 20 randomly sampled character classes for the test set. We train about $39,200$ episodes (or updates).

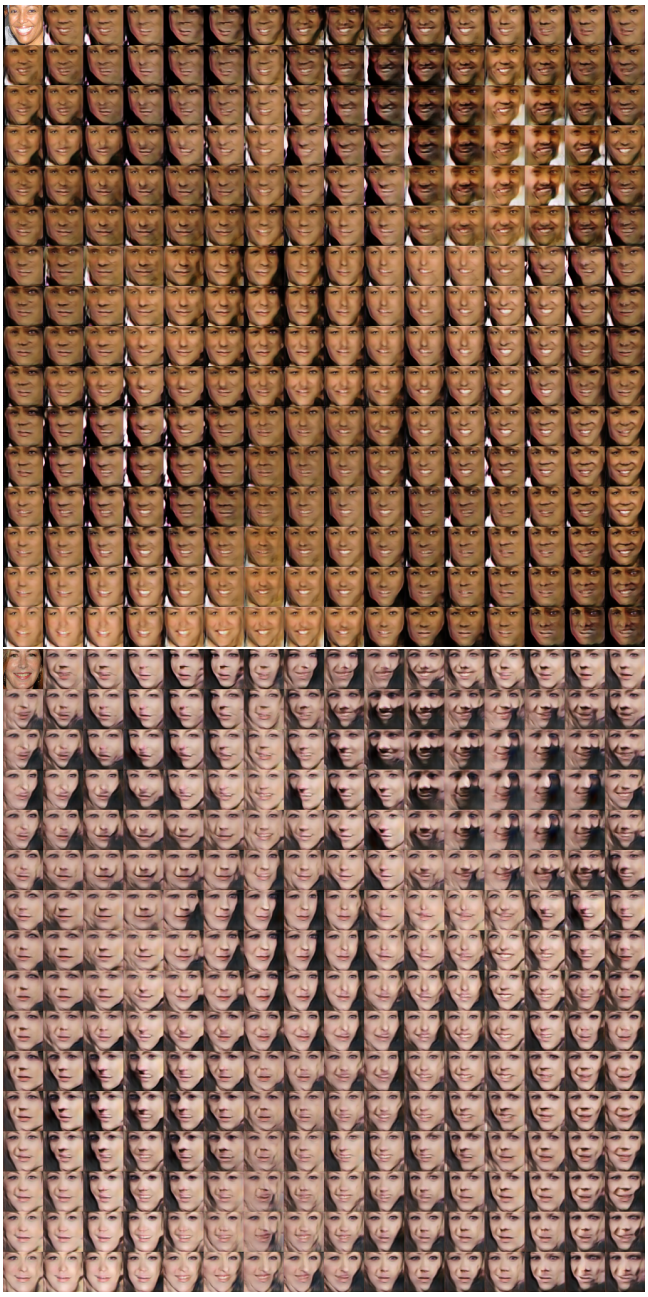
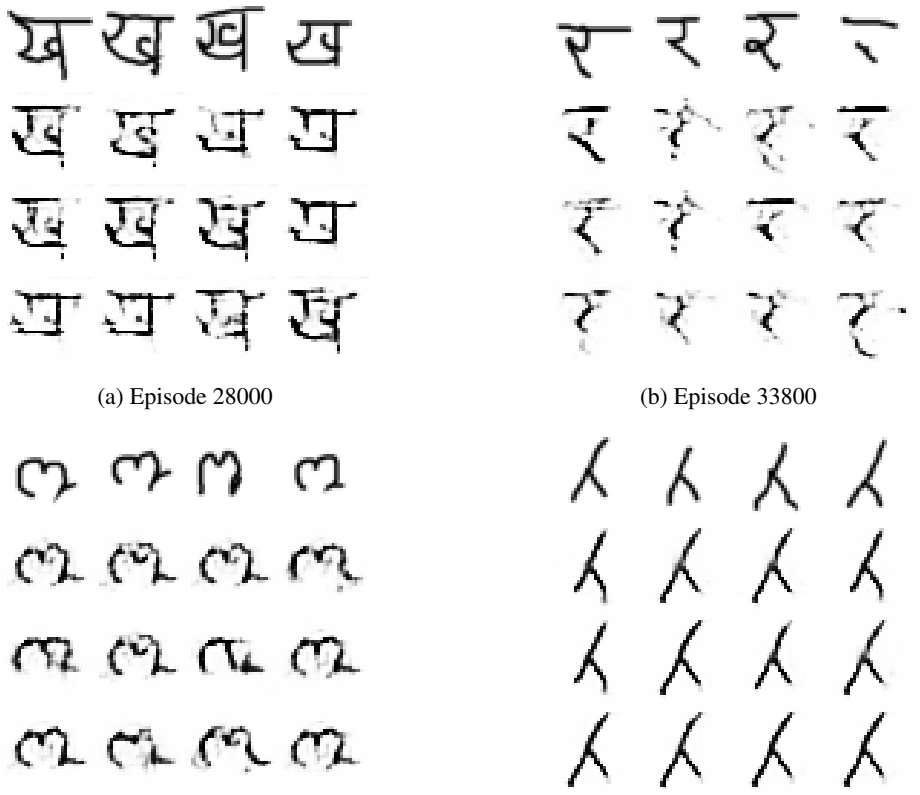


Figure 10: two samples of faces generated by DAGAN

On simpler Omniglot characters like the one shown in Figure 11 (a), the model converges to good results quickly. On more complex characters, the images still have some spots. This is pictured in Figure 11.



(a) Episode 28000

(b) Episode 33800

(c) Episode 35400

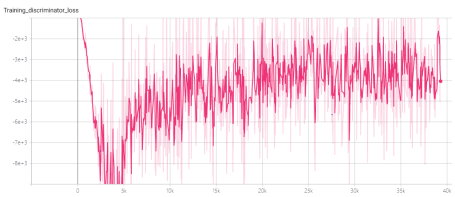
(d) Episode 39200

Figure 11: Generated images of dataset Omniglot

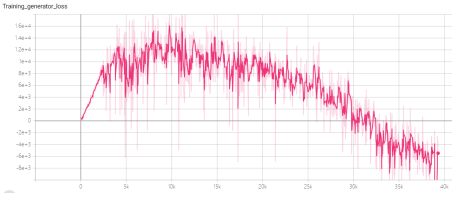
The losses of discriminator and generator are shown in Figure 12. It seems that the model trains not enough and does not converges. The generator loss has a trend to decrease.

FIGR-8 The training classes were all $18,409$ classes, minus 50 randomly sampled classes for the test set. Arguably none of the generated images pictured in Figure 13 can fool a human. However, the images are barely recognizable and show some key features of the images, such as human heads or arrows.

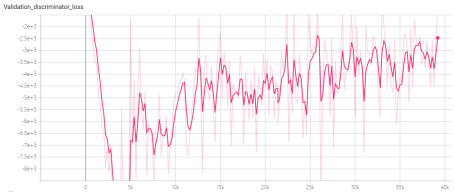
The losses of discriminator and generator are shown in Figure 14. We can see that the discriminator loss fluctuates violently, and the generator loss increases and decreases slowly.



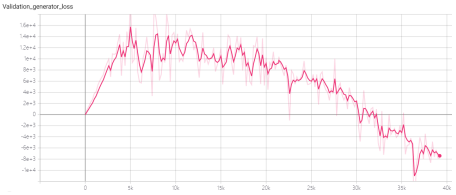
(a) Training discriminator loss



(b) Training generator loss



(c) Validation discriminator loss



(d) Validation generator loss

Figure 12: Losses of Omniglot dataset

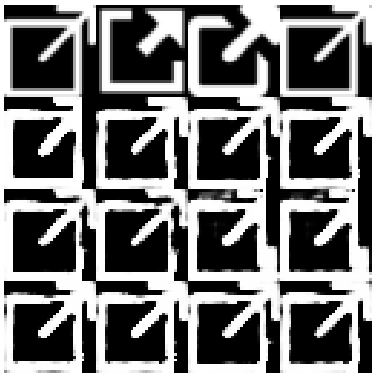
5 Conclusion

We have shown some methods for few-shot image generation, including F2GAN, DA-GAN, and FIGR. All these methods are proposed in recent years and have a considerable improvement in the few-shot learning problem.

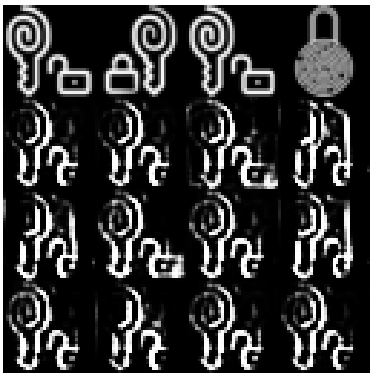
We generate images on Omniglot, VGG-Face, and FIGR-8 datasets. These datasets cover different kinds of images, such as handwritten characters, patterns, and human faces.

By visual comparison, these models have a good performance on simple patterns or characters. However, on the generation problem of complex images, such as the face, there is still a lot of improvement space. According to the quantitative evaluation result, F2GAN has higher IS scores and lower FID scores.

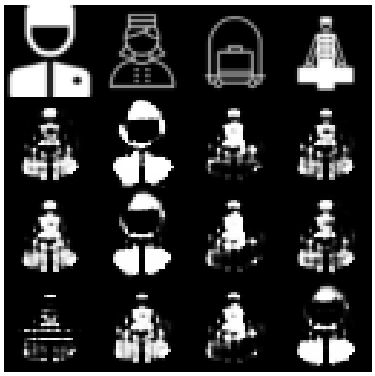
Some further work can be done in the future, such as doing more training epochs in the experiment above to better model and optimize the model structure.



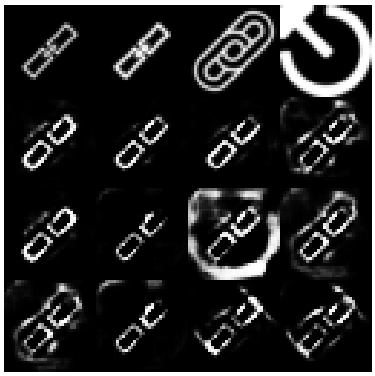
(a) Episode 44000



(b) Episode 100000

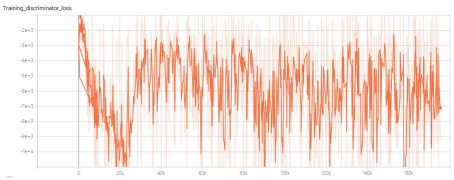


(c) Episode 132600

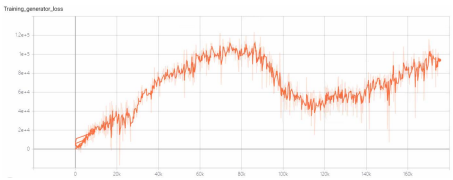


(d) Episode 175200

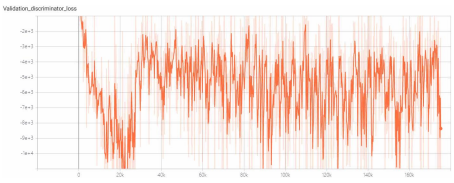
Figure 13: Generated images of dataset FIGR-8



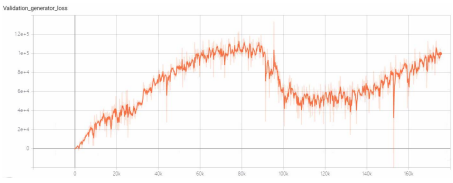
(a) Training discriminator loss



(b) Training generator loss



(c) Validation discriminator loss



(d) Validation generator loss

Figure 14: Losses of FIGR-8 dataset

References

[1] Sergey Bartunov and Dmitry Vetrov. Few-shot generative modelling with generative matching networks. In *International Conference on Artificial Intelligence and Statistics*, pages 670–678, 2018.

[2] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5659–5667, 2017.

[3] Louis Clouâtre and Marc Demers. Figr: Few-shot image generation with reptile. *arXiv preprint arXiv:1901.02199*, 2019.

[4] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *NeurIPS*, 2014.

[6] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. *ArXiv e-prints*, 2017.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] 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. *Advances in neural information processing systems*, 30:6626–6637, 2017.

[9] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017.

[10] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. *arXiv:1608.06993*, 2016.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012.

[12] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[13] Brenden M Lake, Russ R Salakhutdinov, and Josh Tenenbaum. One-shot learning by inverting a compositional causal process. *Advances in neural information processing systems*, 26:2526–2534, 2013.

[14] Stéphane Lathuilière, Enver Sangineto, Aliaksandr Siarohin, and Nicu Sebe. Attention-based fusion for multi-source human image generation. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 439–448, 2020.

[15] Yijun Li, Richard Zhang, Jingwan Cynthia Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. *Advances in Neural Information Processing Systems*, 33, 2020.

[16] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957, 2018.

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

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[19] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[20] Cheng Wang, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Mancs: A multi-task attentional network with curriculum sampling for person re-identification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 365–381, 2018.

[21] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[22] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020.

[23] Huijuan Xu and Kate Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, pages 451–466. Springer, 2016.

[24] Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv:1806.07755*, 2018.

[25] Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Q. Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *CoRR*, abs/1806.07755, 2018.

[26] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pages 7354–7363. PMLR, 2019.

[27] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *CoRR*, abs/1801.03924, 2018.