FEW-SHOT GENERATIVE MODEL WITH FEATURE INTERPOLATION

Shuai Luo(120033910079), Cheng Fei(120033910038), Fan Zhang(119033910126)

Shanghai Jiao Tong University

ABSTRACT

Humans learn new concepts with very little supervision, yet our best deep learning systems need hundreds of thousands of examples. Inspired by recent advances in few-shot learning, we bring dimensionality reduction techniques into this field, we extend the Matching-Based few-shot generative architecture with DrLIM [1], which translates high dimensional data to a low dimensional representation such that similar input objects are mapped to nearby points on a manifold, in this way we obtain a better result in the matching procedure.

***Index Terms*—** Few-shot learning, generative adversarial network

1. INTRODUCTION

Few-shot learning is the problem posed through learning new skills and abilities for tasks from small amounts of labelled data, while few-shot generation is the up-stream of its application. Deep generative networks like Variational Auto-Encoder(VAE) [2] and Generative Adversarial Network [3] have shown excellent performance on generation problems, unfortunately, large quantities of training samples are required, which handicaps its application. Various network architectures have been developed over the years to take on this challenge of few-shot generation, such as Generative Matching Network(GMN) [4]; Data Augmentation Generative Adversarial Network(DAGAN) [5]; Domain Adaptive Few-shot Generation Framework for GANs(DAWSON) [6]; MatchingGAN [7].

The goal of few-shot generation is to build an generative model by pre-training on multiple source domains, the pre-trained model should generate samples with a limited number of examples in the target category provided. MatchingGAN [7], as well as GMN [4], uses encoders to generate the representation vector of the conditional images, and trains it with the weighted reconstruction loss, feature matching loss and other two loss functions, both of them are actually measurements of the difference between the output image and the conditional images or their linear combination. Let us dive deeper, in the matching procedure, a random vector and the conditional images are mapped into the matching space using encoders implemented as neural networks, and then similarity score is computed. In order to maintain the similarities in the original space, we apply contrastive loss function proposed in DrLIM [1] to the training of the encoder, thus we propose a feature interpolation based few-shot generative model in this report.

2. RELATED WORK

Generative Adversarial Networks(GAN) [3], and specifically Deep Convolutional GANs(DCGAN) [8] are extensively used in the field of natural language processing, audio generation and image generation, many improved versions have been proposed in the past years. GANs have emerged as one of the dominant approaches for generating new realistically looking samples. However, while very powerful, GANs can not be applied to situations where acquiring training data is really expensive or even impossible.

Inspired by the matching networks for one-shot learning in discriminative tasks, Generative Matching Network [4] is proposed, a new observation is generated by first mapping both latent vector and conditional images to the same matching space, and then attention kernel is calculated, which provides normalized weights assigned to each conditional observations. Based on the fused features of the conditional objects, generator is trained to output a new observation. Hence forth many networks share the same idea, DAGAN [5] combining generator and discriminator, adversarial training leads the network to generate new images from old ones that are similar enough to be considered within the same category, while dissimilar enough to be a different sample. DAWSON [6] incorporates GANs and MAML-style meta-learning algorithms. MatchingGAN [7] proposed several useful loss functions that are adopted for the model. In this work, we propose a novel network architecture which combines matching procedure, dimensionality reduction techniques and GANs.

3. OUR METHOD

3.1. Problem Statement

We first clarify some terminologies that will be used throughout the discussion of our model. We use $\mathcal{C}_s = \{c_i|_{i=1}^{K}\}$ to denote the collection of seen categories. The collection of unseen categories is denoted as $\mathcal{C}_u = \{c_i|_{i=1}^{M}\}$. Our model Fig. 1 aims to learn a mapping from the conditional images

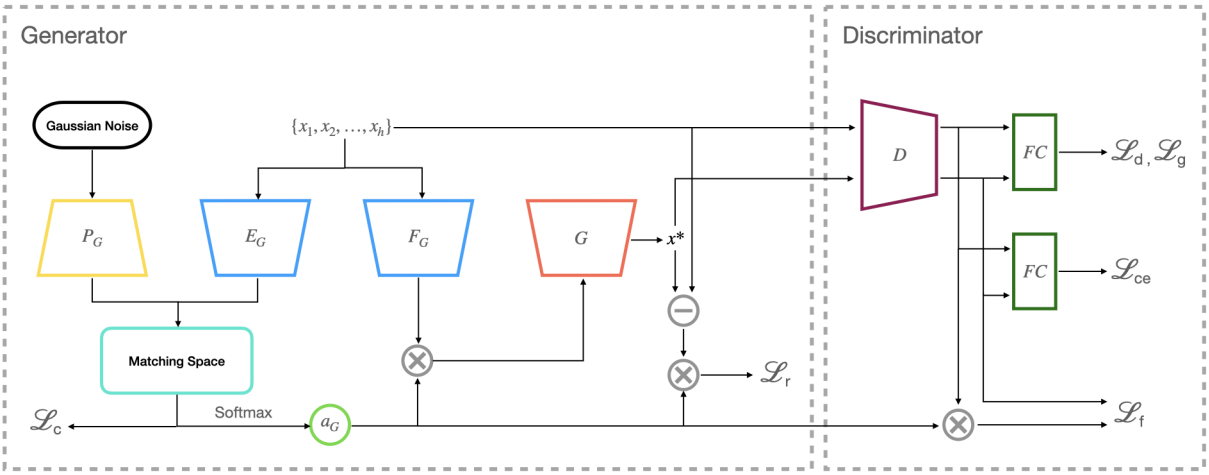

Fig. 1. The framework consists of a generator and discriminator. Gaussian noise and conditional images $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_h\}$ are mapped to the matching space, specifically, contrastive loss $\mathcal{L}_c$ is used for the training of the encoder E_G.

within a category $\mathcal{T} = \{\boldsymbol{x}_i|_{i=1}^h\}$ to a new image $\tilde{\boldsymbol{x}}$ in the same category. During the training, only images from the seen categories are fed into the network. The trained model should be able to output satisfactory image without fine-tuning on the target category.

3.2. Generator

Firstly we sample random noise from the prior $\boldsymbol{z} \sim p(\boldsymbol{z})$, here the prior is simply a standard Normal distribution. Latent vector and conditional observations typically have very different representations, we first project them to the same matching space using P_G and E_G respectively. The problem is to find a function that maps high dimensional input variables to one dimensional outputs, given neighborhood relationships between samples in input space. Here we define the contrastive loss. Unlike loss functions that sum over samples, this loss runs over pairs of samples. Let $\boldsymbol{x}_1, \boldsymbol{x}_2$ be a pair of input vectors, binary label y is defined as,

$$y = \begin{cases} 0, \ \boldsymbol{x}_1, \boldsymbol{x}_2 \in c_i \\ 1, \ \boldsymbol{x}_1 \in c_i, \boldsymbol{x}_2 \in c_j, i \neq j \end{cases} \tag{1}$$

Let $D_{\mathcal{W}}$ denotes the Euclidean distance between the outputs of E_G,

$$D_{\mathcal{W}}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \| \ E_G(\boldsymbol{x}_1) - E_G(\boldsymbol{x}_2) \ \|_2 \tag{2}$$

Where $\mathcal{W}$ is the parameters of the encoder E_G, then the contrastive loss can be written as,

$$\mathcal{L}_c(\mathcal{W}) = \sum_{i=1}^{P} \mathcal{L}_c(\mathcal{W}, (\boldsymbol{x}_1, \boldsymbol{x}_2, y)^i) \tag{3}$$

$$\mathcal{L}_c(\mathcal{W}, (\boldsymbol{x}_1, \boldsymbol{x}_2, y)^i) = (1-y)\mathcal{L}_1(D_{\mathcal{W}}^i) + y\mathcal{L}_2(D_{\mathcal{W}}^i) \tag{4}$$

$(\boldsymbol{x}_1, \boldsymbol{x}_2, y)^i$ is the i-th labeled sample pair, and P is the number of training pairs. Here, we further define $\mathcal{L}_1$ and $\mathcal{L}_2$ as follows,

$$\mathcal{L}_1(D_{\mathcal{W}}) = \frac{1}{2}(D_{\mathcal{W}})^2 \tag{5}$$

$$\mathcal{L}_2(D_{\mathcal{W}}) = \frac{1}{2}\{max(0, m - D_{\mathcal{W}})\}^2 \tag{6}$$

$m > 0$ can be considered as a threshold, only those dissimilar pairs with distance within m will contribute to the loss. In this way, we pull similar pairs together and push dissimilar pairs apart by minimizing $\mathcal{L}_c(\mathcal{W})$ with respect to $\mathcal{W}$. It is obvious that we should form the labeled training set before anything happens, we achieve this by pairing each sample $\boldsymbol{x}_i$ with all other training samples and label the pairs so that $y = 0$ if they are from the same category, and $y = 1$ otherwise. Combining all the pairs, we have $P = C_N^2 = \frac{N!}{2!(N-2)!}$ training cases.

In the matching space, attention kernel is define as,

$$a_G(\boldsymbol{z}, \boldsymbol{x}_i) = \frac{exp(sim(P_G(\boldsymbol{z}), E_G(\boldsymbol{x}_i)))}{\sum_{t=1}^{h} exp(sim(P_G(\boldsymbol{z}), E_G(\boldsymbol{x}_t)))} \tag{7}$$

Here we use the cosine similarity as similarity function. The key component of this network is the encoder F_G and decoder G, which is implemented as a combination of UNet and ResNet. The UResNet has a total of 8 blocks. We interpolate the features extracted by F_G,

$$r_G = \sum_{i=1}^{h} a_G(\boldsymbol{z}, \boldsymbol{x}_i) F_G(\boldsymbol{x}_i) \tag{8}$$

Finally, the decoder is provided with r_G and the latent vector $\boldsymbol{z}$, and outputs the generated image $\boldsymbol{x}^*$. To make sure

that the generated image maintain the fused feature of the conditional images, here we define the weighted reconstruction loss,

$$\mathcal{L}_r = \sum_{i=1}^{h} a_G(\boldsymbol{z}, \boldsymbol{x}_i) \parallel \boldsymbol{x}^* - \boldsymbol{x}_i \parallel_1 \tag{9}$$

3.3. Discriminator

The adversarial discriminator is trained to discriminate between the real images and the generated images, two losses are defined here,

$$\begin{aligned} \mathcal{L}_d &= E_{\boldsymbol{x}^*}[max(0, 1 + \mathrm{D}(\boldsymbol{x}^*)] + E_{\boldsymbol{x}_i}[max(0, 1 - \mathrm{D}(\boldsymbol{x}_i))] \\ \mathcal{L}_g &= -E_{\boldsymbol{x}^*}[\mathrm{D}(\boldsymbol{x}^*)] \end{aligned} \tag{10}$$

Discriminator D is trained to minimize $\mathcal{L}_d$, while the generator is trained to minimize $\mathcal{L}_g$. We replace the full connected layer with another FC layer with K outputs, which is the number of seen categories. Then we employ the cross entropy loss,

$$\mathcal{L}_{ce} = -\log p(c(\boldsymbol{x})|\boldsymbol{x}) \tag{11}$$

$c(\boldsymbol{x})$ is the category of image $\boldsymbol{x}$. Lastly, to cooperate with the feature fusion strategy, we employ the feature matching loss,

$$\mathcal{L}_f = \parallel \sum_{i=1}^{h} a_G(\boldsymbol{z}, \boldsymbol{x}_i)\hat{D}(\boldsymbol{x}_i) - \hat{D}(\boldsymbol{x}^*) \parallel_1 \tag{12}$$

3.4. Training

The total loss function can be written as,

$$\mathcal{L} = \lambda_c\mathcal{L}_c + \mathcal{L}_d + \mathcal{L}_g + \lambda_r\mathcal{L}_r + \mathcal{L}_{ce} + \lambda_f\mathcal{L}_f \tag{13}$$

During adversarial learning, the discriminator D is trained with $\mathcal{L}_d$ and $\mathcal{L}_{ce}$, while the generator is trained with $\mathcal{L}_c$, $\mathcal{L}_g$, $\mathcal{L}_r$, $\mathcal{L}_{ce}$, and $\mathcal{L}_f$.

4. EXPERIMENTS

Our team is familiar with pytorch instead of tensorflow, so we've obtained some results based on DAGAN originally proposed by Antreas Antoniou.

We conduct experiments on two datasets: Omniglot and VGGFace. And we choose FIGR and GMN as baselines.

For Omniglot (*resp.*, VGGFace), a total of 1623 (*resp.*, 2395) categories are split into 1200 (*resp.*, 1802) seen categories, 212 (*resp.*, 497) validation seen categories, 211 (*resp.*, 96) unseen categories. Validation seen categories are used to monitor the training procedure, but not engaged in updating model parameters. For VGGFace, some categories have more than 100 samples. For these categories, we randomly choose 100 images from each category to fit a low-data setting.

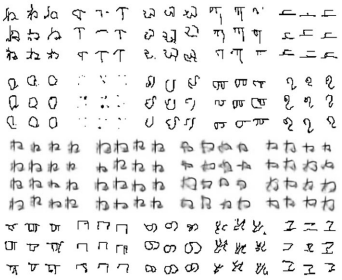

Fig. 2. Images generated by DAGAN on Omniglot.

Fig. 3. Images generated by DAGAN on VGGFace.

4.1. Quantitative Evaluation of Generated Images

We evaluate the quality of images generated by different methods on VGGFace dataset based on commonly used Inception Scores (IS) [9] and Fréchet Inception Distance (FID) [10].

We generate 128 images for each unseen category using each method(FIGR, GMN and DAGAN), based on which FID and IS are calculated. See Table 1.

4.2. Low-data Classification

To further evaluate the quality of generated images, we use generated images to help downstream classification tasks in low-data setting in this section. For low-data classification on unseen categories, we randomly select a few (*e.g.*, 5, 10, 15) training images per unseen category while the remaining images in each unseen category are test images.

We use ResNet18 [11] pretrained on seen categories as backbone network, train the classifier based on the training

Table 1. FID (↓) and IS (↑) of images generated by different methods on VGGFace dataset.

Methods	FID (↓)	IS (↑)
FIGR	154.21	5.19
GMN	201.12	6.38
DAGAN	120.63	3.97

Table 2. Accuracy(%) of different methods on different datasets in low-data setting.

Method	Dataset	Accuracy		
		5	10	15
Standard	Omniglot	66.22	81.87	83.31
FIGR	Omniglot	69.23	83.12	84.89
GMN	Omniglot	67.74	84.19	85.12
DAGAN	Omniglot	87.73	89.30	95.33

images of unseen categories, and finally predict the test images of unseen categories.

We use generated images to augment the training set of unseen categories. For each few-shot generation method, we generate 512 images for each unseen category based on the training set of unseen categories. Then, the ResNet18 classifier is trained on the augmented training set (original training set and generated images) and applied to the test set of unseen categories.

5. CONCLUSION

In our work, we initially proposes to bring dimensionality reduction methods into few-shot generation problem, and we believe that the performance of the MatchingGAN should be improved by combining a new loss function into the original one. However, our teammates used to code under the framework of pytorch instead of tensorflow, so we ran the pytorch implementation of DAGAN on two datasets and obtained some results. The result is very similar to those listed in the paper of MatchingGAN.

F. REFERENCES

[1] R. Hadsell, S. Chopra, and Y. Lecun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2006.

[2] Jianmin Bao, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua, "CVAE-GAN: fine-grained image generation through asymmetric training," in *ICCV*, 2017.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *NeurIPS*, 2014.

[4] Sergey Bartunov and Dmitry Vetrov, "Few-shot generative modelling with generative matching networks," in *ICAIS*, 2018.

[5] Antreas Antoniou, Amos Storkey, and Harrison Edwards, "Data augmentation generative adversarial networks," *arXiv preprint arXiv:1711.04340*, 2017.

[6] Weixin Liang, Zixuan Liu, and Can Liu, "DAWSON: A domain adaptive few shot generation framework," *CoRR*, vol. abs/2001.00576, 2020.

[7] Yan Hong, Li Niu, Jianfu Zhang, and Liqing Zhang, "Matchinggan: Matching-based few-shot image generation," in *2020 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2020, pp. 1–6.

[8] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *Computer ence*, 2015.

[9] 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.

[10] 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 *NeurIPS*, 2017.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[12] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al., "Matching networks for one shot learning," in *NeurIPS*, 2016.

[13] 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 *CVPR*, 2018.

[14] Chelsea Finn, Pieter Abbeel, and Sergey Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.

[15] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele, "Meta-transfer learning for few-shot learning," in *CVPR*, 2019.

[16] Wenbin Li, Lei Wang, Jinglin Xu, Jing Huo, Yang Gao, and Jiebo Luo, "Revisiting local descriptor based image-to-class measure for few-shot learning," in *CVPR*, 2019.