FINAL REPORT

Shengyuan Huang, Ledian Liu, Ling Dai

1. INTRODUCTION

Humans can learn new concepts (object classes) from very few samples efficiently. For example, children who have seen cats and dogs only a few times can quickly distinguish them. However, well-trained machine learning models, especially those with deep structures and huge amounts of parameters, require a large number of samples to achieve the similar performance of humans. This motivates the setting of few-shot learning whose goal is to classify or generate new data having seen only a few training examples. In practice, few-shot learning is useful when training examples are hard to find, or when the cost of labelling data is high.

Deep learning has made unprecedented achievements in areas such as vision, language and speech, but is still suffering from requiring large datasets. In low-data regime, deep nerual networks usually overfit on the training set and producing poor generalization on the test set. Although, techniques have been developed to alleviate the overfitting problem, such as regularization loss, batch normalization, batch renormalization and dropout. They work poorly in few-shot setting, since the flexibility of the networks is too high and the training set is too small.

In few-shot setting, we need an approach to learn knowledge that is shared by all classes, or at least a collection of classes, so that the approach can still work well when encountering new classes. It is also possible to generate more data from existing data without changing class labels, by applying various transformations to the original dataset. These transformations, such as random translations, rotations and flips as well as addition of Gaussian noise, are valid for data from all classes. There are two properties for this kind of transformations: (a) preserving the class labels, (b) applicable to all classes. We define the transformation that satisfy these two properties as class-agnostic transformation. However, the transformations mentioned above only take a very small part of the class-invariant transformation space. Hence, in this paper, we propose Class-Agnostic Transformation Generative Adversarial Network (CAT-GAN), which can apply a large variety of class-agnostic transformations to the input image leveraging the powerful expressive ability of DNNs. To improve the diversity of the model, we introduce a new reconstruction loss.

2. RELATED WORK

2.1. Data Augmentation

Data Augmentation is widely used in classification tasks. It is difficult to encode known invariances into model parameters [1]. However, it can be easier to encode those invariances in the data by generating new samples through transformations from existing samples. For example the labels of handwritten characters should be invariant to small shifts in location, small rotations, changes in intensity, changes in stroke thickness, changes in size etc. Almost all kinds of data augmentation are based on priori knowledge. Among papers that try to learn data augmentation strategies, the work of [2] is worthy of note, where the authors learn augmentation strategies on a class by class basis. Nevertheless, their method cannot work in few-shot setting, since unseen classes are encountered.

2.2. Generative Adversarial Networks (GAN)

GANs are an exciting recent innovation in machine learning [3, 4]. GANs are generative models: they create new data instances that resemble your training data. For example, GANs can create images that look like photographs of human faces, even though the faces don't belong to any real person. Recent improvements in the optimization process have reduced some of the failure modes of the GAN learning process such as [5]. To enable stable training, an alternative to clipping weights is provided: penalize the norm of gradient of the critic with respect to its input [6].

2.3. few-shot learning

Few-shot learning is a challenging problem and is receiving significant attention in recent years. In [7], modern deep learning architecture have been used for one-shot conditional generation. They use a sequential generative model to achieve one-shot generation. The inference process uses an attention module to have a Variational Auto Encoder attend to a section of the generated image sequentially. It generates binary images of size 28×28 and 52×52 on the Omniglot dataset with one-shot learning. A different approach uses matching networks to achieve few-shot image generation [8]. In essence, matching networks are memory-assisted networks that leverage an external memory by employing an attention module

to quickly learn new concepts. It assumes that the concepts stored are somewhat similar to the new concepts.

3. OUR METHODS

Before introducing our method, we first describe some notations. For ease of representation, in the rest of this paper, we use a bold (*resp.*, plain) letter to denote a non-scalar (*resp.*, scalar), and use $\mathbf{x}^T$ to denote the transpose of a vector $\mathbf{x}$. We use $\mathcal{D} = \{(\mathbf{I}_i, \mathbf{y}_i)|_{i=1}^{N_I}\}$ to denote our image dataset, in which $\mathbf{I}_i$ is the i-th image with one-hot label vector $\mathbf{y}_i$ and N_I is the number of images. We use $\mathcal{D}^c = \{(\mathbf{I}_i^c, c)|_{i=1}^{N_I^c}\}$ to denote the samples labeled with c in the image dataset, in which $\mathbf{I}_i^c$ is the i-th image in the class c and N_I^c is the number of class c in the dataset.

Intuitively, if we have a transform function for data, which can keep its class label invariant, we can use this transformation function to generate additional data, especially for those few-shot data. Though we don't know what transform function can meet our expectation, we can attempt to learn a valid transformation function from those related problems that we can apply to our setting. Based on this idea, we take Generative Adversarial Network(GAN) as the basic framework and propose Class-agnostic Transformation Generative Adversarial Network(CAT-GAN), a novel method to generate new data from limited data.

As is shown in Figure 1, our CAT-GAN consists of four components: CNN encoder G_E to project the original image and latent code into a vector, CNN decoder G_D to decode the vector back into an image, discriminator D to distinguish fake images from real ones, and regressor R to recover the latent code from the generated image. Then we will introduce how our model works in detail.

Given an image $\mathbf{I}_i^c$ in $\mathcal{D}^c$, we take $\mathbf{I}_j^c$ in the same $\mathcal{D}^c$ randomly, which means $\mathbf{I}_i^c$ and $\mathbf{I}_j^c$ belong to the same class. We sample a latent code $\mathbf{z}_i$ from unit Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$, which aims to get diverse results for our model. Then we aggregate $\mathbf{I}_i^c$ and $\mathbf{z}_i$ with a certain method and put them through G_E to get a feature vector $\mathbf{f}_i^c$, which can be written as:

$$\mathbf{f}_i^c = G_E(\mathbf{I}_i^c, \mathbf{z}_i) \tag{1}$$

After obtaining feature vector $\mathbf{f}_i^c$, which containing the information of the input image $\mathbf{I}_i^c$ and the latent code $\mathbf{z}$, we use G_D to decode the feature vector $\mathbf{f}_i^c$ and generate a new image $\hat{\mathbf{I}}_i^c$, which can be expressed as:

$$\hat{\mathbf{I}}_i^c = G_D(\mathbf{f}_i^c) \tag{2}$$

The generated image $\hat{\mathbf{I}}_i^c$ contain the information of $\mathbf{z}$, and we hope we can recover $\mathbf{z}$ from $\hat{\mathbf{I}}_i^c$. To achieve this, we design a regressor R to reconstruct $\mathbf{z}$. And then we use L_2 loss to encourage a bijection between the reconstructed code $\hat{\mathbf{z}}$ and latent code $\mathbf{z}$, which can alleviate mode collapse problem and help produce more diverse images. These can be written as:

$$\hat{\mathbf{z}}_i = R(\hat{\mathbf{I}}_i^c) \tag{3}$$

$$L_R = \frac{1}{N_I} \sum_c \sum_{i=1}^{N_c^I} \|\mathbf{z}_i - \hat{\mathbf{z}}_i\|_2^2 \tag{4}$$

Then we use discriminator D to distinguish generated images from the real ones by maximizing the following adversarial loss:

$$L_D = \frac{1}{N_\mathbf{I}} \sum_c \sum_{i=1}^{N_c^I} [\log(D(\mathbf{I}_i^c, \mathbf{I}_j^c)) + \log(1 - D(\hat{\mathbf{I}}_i^c, \mathbf{I}_j^c))] \tag{5}$$

At last, we combine two losses and train our model. The full loss function of our CAT-GAN can be written as:

$$L_{all} = \min_{G_E, G_D, R} \max_D \lambda_R L_R + \lambda_D L_D \tag{6}$$

in which λ_R and λ_D are hyper-parameters. The objective in Equatioin (6) can be optimized by updating $\{G_E, G_D, R\}$ and $\{D\}$ in an alternating manner.

In the testing stage, we ignore R and D and only reserve G_E and G_D. For an input image, we sample several latent codes from unit Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$ and then pass the image and the latent codes through G_E and G_D, which can lead to several generated images whose class label is the same as the input image.

4. EXPERIMENTS

4.1. Dataset

4.1.1. Omniglot dataset

The Omniglot data[9, 10] set is designed for developing more human-like learning algorithms. It contains 1623 different handwritten characters from 50 different alphabets. Each of the 1623 characters was drawn online via Amazon's Mechanical Turk by 20 different people. Each image is paired with stroke data, a sequences of [x,y,t] coordinates with time (t) in milliseconds. We will conduct three tasks to evaluate one-shot generalization as in [9] to test the generalization capability of our algorithm. The three tasks are: **1)unconditional (free) generation**: unconditional refers to generating samples unconditionally from the dataset, **2) generation of novel variations of a given exemplar:** At test time, the model is presented with a character of a type it has never seen before (was not part of its training set), and asked to generate novel variations of this character.The context x' is the image that we wish the model to generate new exemplars of. To expose the boundaries of our approach, we test this under weak and strong one-shot generalization tests. and **3) generation**

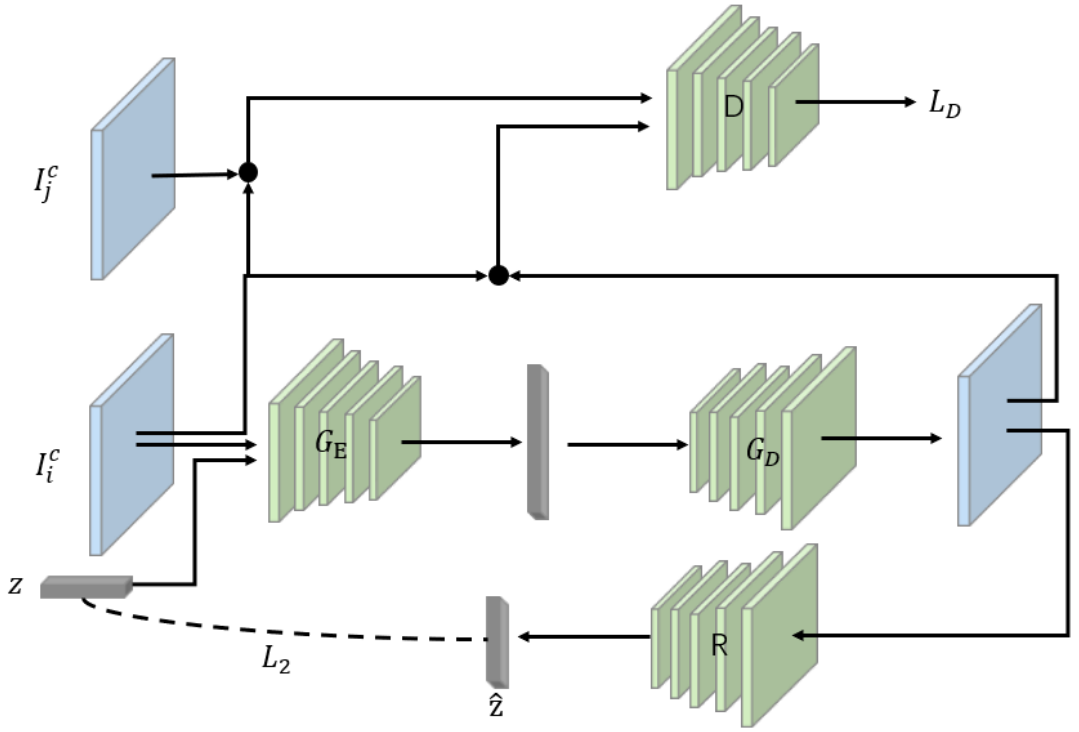

Fig. 1. Our CAT-GAN architecture. G_E takes an image $\mathbf{I}_i^c$ and a latent code $\mathbf{z}$ as input and output an encoded feature $\mathbf{f}_i^c$. G_D takes $\mathbf{f}_i^c$ as input and generates an image $\hat{\mathbf{I}}_i^c$. R takes $\hat{\mathbf{I}}_i^c$ as input and reconstruct the latent code $\mathbf{z}$. D takes $\mathbf{I}_i^c$, $\mathbf{I}_j^c$ and $\hat{\mathbf{I}}_i^c$ as input and discriminate the real image from fake one.

of representative samples from a novel alphabet This task conditions the model on any number between 1 to 10 samples of a novel alphabet and asks the model to generate new characters consistent with this novel alphabet. We will test on the hardest condition form of this test, using only 1 context image.

The Omniglot data was resized to 32×32 and 64 × 64. The training classes where all 1623 characters in the dataset minus 20 randomly sampled character classes for the test set.

4.1.2. VGG-Face dataset

The VGG-Face dataset [11] contains 982,803 images from 2,622 celebrities spanning a wide range of ethnicities and professions. The Images were collected from Google Image Search with large variations in pose, age, lighting, and background. The dataset is approximately gender-balanced, achieved by selecting the same candidates in the data collection stage. The number of images for each identity ranges from 80 to 843, with an average of 374 images per identity. VGG-Face provides a large-scale training dataset of depth, which has a limited number of subjects but many images for each subject. The depth of the dataset enforces the trained model to address a wide range of intraclass variations, such as lighting, age, and pose. However, like other large-scale datasets, Vgg-Face is constructed by scraping websites like Google Images and celebrities on formal occasions: smiling, makeup, young, and beautiful. They are largely different from databases captured in daily life. The biases can be attributed to many exogenous factors in data collection, such as cameras, lightings, preferences over certain types of backgrounds, or annotator tendencies. Dataset biases adversely affect cross-dataset generalization.

4.2. Baselines

Data augmentation generative adversarial networks (DAGAN)

DAGAN learns how to generate a synthetic image using a lower-dimensional representation of a real image. Rather than the generator taking as input a class and noise vector, in the DAGAN framework, the generator is essentially an autoencoder: it takes an existing image, encodes it, adds noise, and decodes it. So, the decoder learns a large family of transformations for data augmentation.

The DAGAN discriminator distinguishes between an im-

age and a transformed version on the one hand, and a pair of images from the same class on the other hand. So, the discriminator incentivizes the decoder to learn transformations which do not change the class, but which are non-trivial in the sense that the transformed image is not too similar to the original image. However, a key assumption of the DAGAN is that the same transformations apply to all classes — this is reasonable in the computer vision context, but less so in fraud or anomaly detection.

4.3. Evaluation metrics

We use classifier accuracy to evaluate our method by comparing the results of using different image augmentation methods. For classification on unseen categories, we randomly select a few training images per unseen category while the remaining images in each unseen category are test images. Note that we have training and testing phases for the classification task, which are different from the training and testing phases of our CAT-GAN. We use the images of seen categories, and then train the classifier using the training images of unseen categories. Then, the trained classifier is used to predict the test images of unseen categories.

On the other hand, we use the generated images to augment the training set of unseen categories. For each few-shot generation method, we generate some images for each unseen category based on the training set of unseen categories. Then, we train the classifier on the augmented training set (including both original training set and generated images) and apply the trained classifier to the test set of unseen categories.

4.4. Implementation details

The structure of our generator G is similar to DAGAN [12], which is a combination of a UNet and ResNet. G has a total of 8 blocks, with each block having 4 convolutional layers (with leaky rectified linear (relu) activation function and batch renormalization (batchrenorm) [13]) followed by one downscaling or upscaling layer. Downscaling layers (in blocks 1-4) are convolutions with stride 2 followed by leaky relu, batch normalisation and dropout. Upscaling layers were stride 1/2 replicators, followed by a convolution, leaky relu, batch renormalization and dropout. For Omniglot experiments, all layers had 64 filters. For the VGG-Faces the first 2 blocks of the encoder and the last 2 blocks of the decoder had 64 filters and the last 2 blocks of the encoder and the first 2 blocks of the decoder 128 filters. In addition each block has skip connections. As with a standard ResNet, a strided $1x1$ convolution also passes information between blocks, bypassing the between block non-linearity to help gradient flow. Finally skip connections were introduced between equivalent sized filters at each end of the network (as with UNet).

For our discriminator D, we use a DenseNet [14] discriminator, using layer normalization instead of batch normalization; the latter will break the assumptions of the WGAN objective function. The DenseNet is composed of 4 Dense Blocks and 4 Transition Layers, as they are defined in [14]. We use a growth rate of $k = 64$ and each Dense Block had 4 convolutional layers within it. We also used dropout at the last convolutional layer of each Dense Block as we find that this improves sample quality.

For classification experiments we use a DenseNet classifier composed of 4 Dense Blocks and 4 Transition Layers with a $k = 64$, each Dense Block has 3 convolutional layers within it. The classifiers are a total of 17 layers (i.e. 16 layers and 1 softmax layer). Furthermore we apply a dropout of 0.5 on the last convolutional layer in each Dense Block.

For hyper-parameter, we set the dimension of noise z to 100. And we set $\lambda_R = 0.1$ and $\lambda_D = 1$ in equation (6).

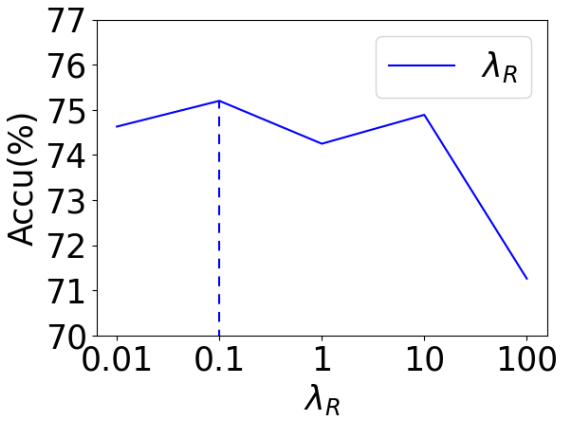

Fig. 2. Analyses of hyper-parameters λ_R. The default values are indicated by vertical dashed lines. We use Omniglot dataset on the setting of 10-samples.

4.5. Experiment results

To evaluate the quality of our generated images, we use generated images to help classification tasks. For classification on unseen categories, following MatchingGAN [15], 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. Note that we have training and testing phases for the classification task, which are different from the training and testing phases of our CAT-GAN. We train the classifier using the training images of unseen categories. Finally, the trained classifier is used to predict the test images of unseen categories. This setting is referred to as "Standard" in Table 1.

Then, we use the 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, we train the classifier on the augmented training set (including

Dataset	Omniglot			VGGFace		
Number of samples	5-samples	10-samples	15-samples	5-samples	10-samples	15-samples
Standard	65.12	80.79	82.13	8.41	19.56	37.45
DAGAN [12]	71.68	73.75	74.51	17.96	32.63	42.17
CAT-GAN	**73.46**	**75.20**	**76.11**	**20.01**	**34.10**	**43.57**

Table 1. Accuracies(%) of different methods on two datasets Omniglot and VGGFace. The best results are denoted in boldface.

both original training set and generated images) and apply the trained classifier to the test set of unseen categories.

The experiment results are shown in Table 1. From the table, we can see that with the increase of number of samples, the accuracy increases. The result of DAGAN [12] is better that standard. And our CAT-GAN shows the best performance.

4.6. Hyper-parameter Analyses

We attempt to study the affect of hyper-parameters on the setting of Omniglot dataset and 10-samples. We vary λ_R in Eqn. (6) in the range of $[0.01, 100]$. The results are plotted in the Fig. 2, which demonstrates that our method is robust with λ_R in a reasonable range.

4.7. Visualization of generated images

In this section, we show the generated images in Fig. 3. The first column shows the input images. For each input image, we show three images generated by DAGAN (shown in the next three columns in Fig. 3) and three images generated by CAT-GAN (shown in the last three columns in Fig. 3). The first three input images is in VGGFace dataset and the rest three input images is in Omniglot dataset. We can see that the performance of our CAT-GAN is better than DAGAN.

5. CONCLUSION

In this paper, we propose a GAN-based augmentation method CAT-GAN, which is trained on images from seen categories and applies class-agnostic transformations to each image from unseen categories. Moreover, we design a novel latent code reconstruction loss to improve the diversity of generated images. Finally, we demonstrate that CAT-GAN can generate realistic and diverse images, and also improve performance of classifiers in low-data setting on two datasets.

6. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[2] Søren Hauberg, Oren Freifeld, Anders Boesen Lindbo Larsen, John Fisher, and Lars Hansen, "Dreaming more data: Class-dependent distributions over diffeomorphisms for learned data augmentation," in *Artificial Intelligence and Statistics*, 2016, pp. 342–350.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[4] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[5] Martin Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[6] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.

[7] Danilo Jimenez Rezende, Shakir Mohamed, Ivo Danihelka, Karol Gregor, and Daan Wierstra, "One-shot generalization in deep generative models," *arXiv preprint arXiv:1603.05106*, 2016.

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

[9] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum, "Human-level concept learning through probabilistic program induction.," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[10] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum, "The omniglot challenge: a 3-year progress report," *Current opinion in behavioral sciences*, vol. 29, pp. 97–104, 2019.

[11] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman, "Deep face recognition," in *British Machine Vision Conference*, 2015.

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

[13] Sergey Ioffe, "Batch renormalization: Towards reduc-

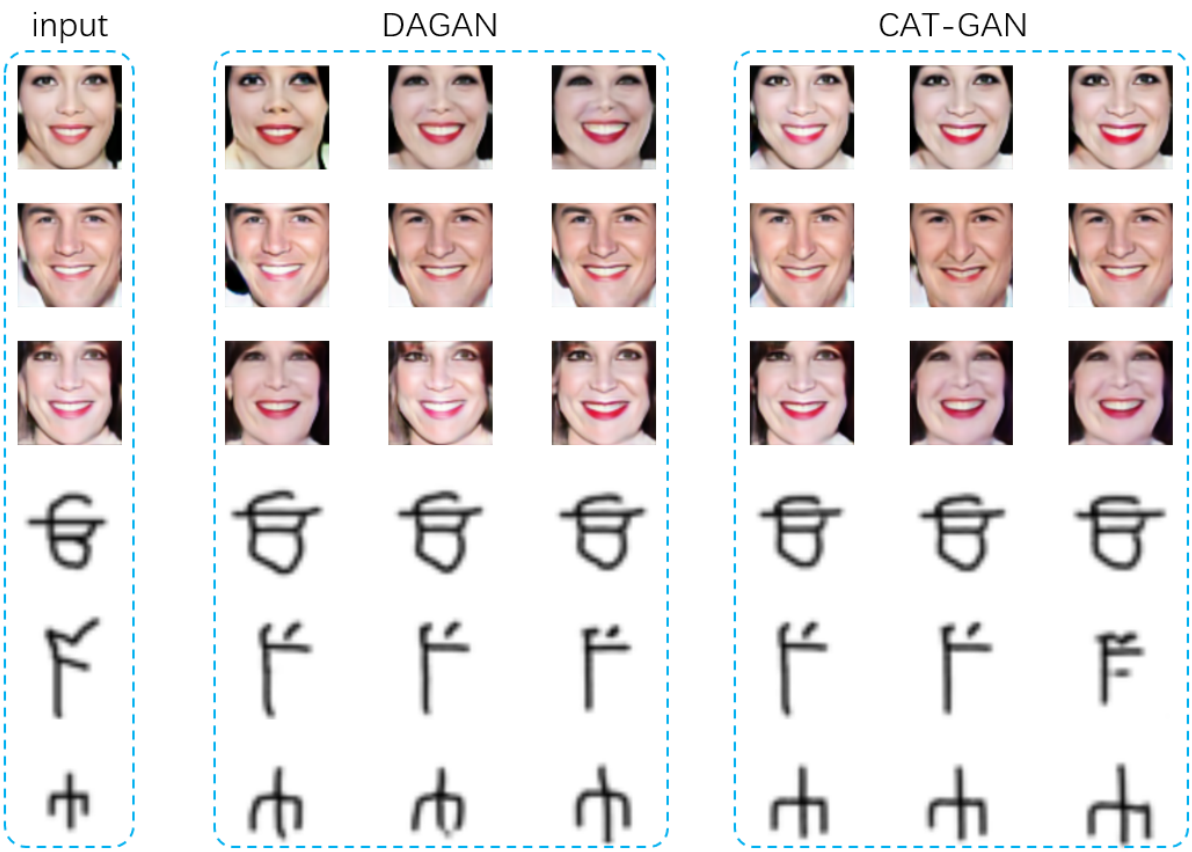

Fig. 3. Images generated by DAGAN and CAT-GAN on two datasets Omniglot and VGGFace. The first column is the input image. For each input image, we show three images generated by DAGAN and three images generated by CAT-GAN. The first three input images is in VGGFace dataset and the rest three input images is in Omniglot dataset.

ing minibatch dependence in batch-normalized models," *Advances in neural information processing systems*, vol. 30, pp. 1945–1953, 2017.

[14] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[15] 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.