Personalized Data Generation using GAN

Yang Fang
Department of Electrical & Computer Engineering
University of Wisconsin-Madison
NetID: fang65
fang65@wisc.edu

Prof. Kangwook Lee, Prof. Matthew Malloy
Project Advisor
Department of Electrical & Computer Engineering
University of Wisconsin-Madison

1 Abstract

Personalized data generation is to fill missing data in the dataset with customized requirements. This project dives into the application of Generative Adversarial Networks(GAN) to solve the nonlinear matrix completion problem. In this project, because of the excellent performance of clustering and GAN, we proposed a novel cluster-GAN-cluster approach towards Personalized Data Generation. For the clustering part, we created an interesting mapping strategy to feed the GAN. We have done the the qualitative evaluation by showing convincing sample outputs by testing on the dataset of FEMNIST. The feasibility of using the method of cluster-GAN-cluster has also been proved by testing. After Kmeans Clustering, the generated image from DCGAN will have high possibility (at least 80%) to still be clustered in the same cluster where the training dataset is in.

2 Topic Introduction and Background

Generating convincing personalized data is a problem that is interesting but challenging in image processing and related fields. Especially in the situation of missing some important data, personalized data generation can play an important role here to fill these missing data with customized requirements. There exists some techniques to do data or image generation, however they still have their own limitations due to the randomness and diversity of natural images. Generative Adversarial Network (GAN)[1], proposed in 2014, has already performed attractive results in traditional data generation. How about using GAN to solve this challenging problem?

We will first clarify and simplify this problem as following.

Imaging we have 1000 different users and they all write from digit 0 to digit 9 in different ways. Then we can give the 1000 users an index of i from 0 to 999 respectively. Also, we can respectively give the different digits an index of j from 0 to 9.

The above diagram is to show the matrix of the dataset. Let each image sample in the matrix be $X(i, j)$, which means the sample of user i given digit j. Assume we lost some samples in this dataset and would like to fill them with exact locations given the indexes of both the digits and users.

3 Problem Statement

In this problem, we are given samples of defined domains for different users and our task is to generate new samples with the appointed index of user and digit. This problem of "Personalized

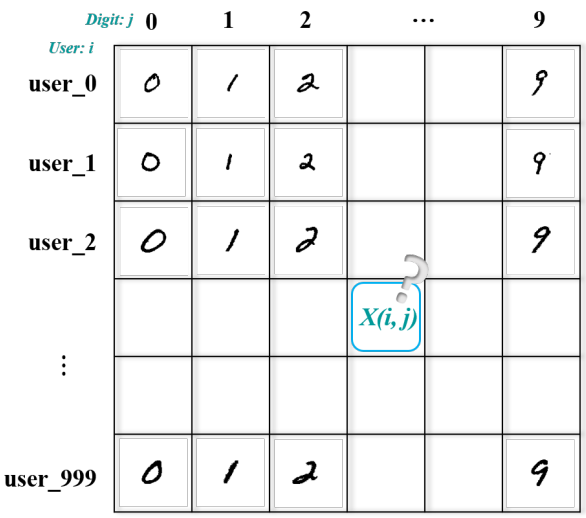

Figure 1: Different handwriting styles of 1000 users. User data is from the dataset of FEMNIST [2]

Data Generation" can be deconstructed as a Nonlinear Matrix Completion problem where each row represents a user and each column represents a domain(digit).

4 Related Research

If the samples in the matrix are numbers instead of images, this problem will be a linear matrix completion problem. From previous lectures we know that this problem is usually solved by Singular Value Decomposition (SVD). However, our problem here is different from pure number matrix problem, the progress of image processing is non-linear here. Therefore, we should try other methods. As we said before, one of the most interesting methods for data generation is GAN. It has been 6 years for GAN to be optimized and improved. From those methods based on GAN, Collaborative Generative Adversarial Network (CollaGAN) [3] is one of the current state-of-the-art methods. This method only use a single generator and discriminator network by combining the corresponding available data in multi-domain to do data estimation. Therefore, we would like to apply this method in our personalized data generation project. The following is a diagram for CollaGAN.

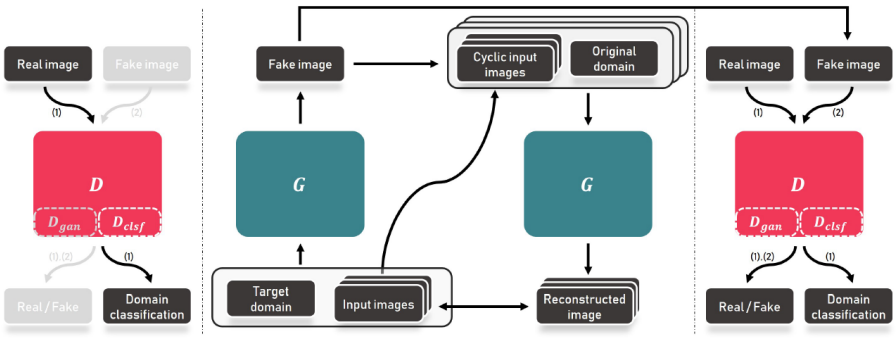

Figure 2: Flowchart for GAN [3]

5 Mathematical Formulation

5.1 Nonlinear Matrix Completion

Matrix Completion can be considered as a prediction problem, where we can view the prediction function as an inner product between the latent codes of users and handwriting in our problem. This latent codes are mapping functions of user-handwriting. Nonlinear matrix completion differs from linear matrix completion in their latent codes. The mapping function of linear matrix completion is a linear one, while the nonlinear matrix completion is a nonlinear one[4].

We can build our nonlinear matrix completion problem as following:

$A(i,x) = \langle \mathcal{U}(i), \mathcal{V}(x) \rangle, i \in \mathcal{I}, x \in \mathcal{X}, \mathcal{U} : \mathcal{I} \to \mathcal{S}, \mathcal{V} : \mathcal{X} \to \mathcal{S}$

where x is one sample image from the dataset, i is the user i, $A(i,x)$ is the possibility that $user_i$ writes the image x, $\mathcal{U}$ is the nonlinear mapping from the user space to the latent code space, $\mathcal{V}$ is the nonlinear mapping from the sample image space to the latent code space.

5.2 Generative Adversarial Network(GAN)

GAN is an outstanding unsupervised machine learning framework in data and image generation.

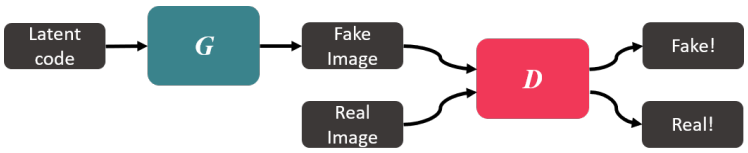

Figure 3: Flowchart for CollaGAN [1]

Usually, a GAN has two parts [1]:

The first one is the Generator G learning part, which uses the input latent code to generate plausible images. The Generator G can also be seen as a mapping from the distribution of latent code z to the distribution of generated image $p(x)$, which can be denoted as $G : \mathcal{Z} \mapsto \mathcal{X}$.

The second one is the Discriminator D learning part. This part has two tasks: distinguish and penalize. Through several training, it learns to distinguish the fake images from the real images in the training dataset and gives penalization to the Generator that generates implausible images. The Discriminator maps the input generated image to the possibility of the sample image being real, which can be denoted as $D : \mathcal{X} \mapsto \mathbb{R}$

The Generator and Discriminator compete with each other through a min-max game so that they can be in their best condition. The Generator G and Discriminator D can be modeled as a convolutional neural network respectively, and we can paramiterize them as Θ_G and Θ_D respectively. We can also denote the prior noise distribution as $P(z)$. Then we can define our game as: $\min_{\Theta_G} \max_{\Theta_D} \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$

6 Project Approach

6.1 Approach 1

Similar to matrix completion, any entry of the matrix (i,j) can be generated using a latent representation of user and digit.

$$\text{Entry: } (i,j) = F(i*, j*)$$

where F is the mapping function,

$i*$ is the latent matrix of user i,

$j*$ is the latent matrix of domain j.

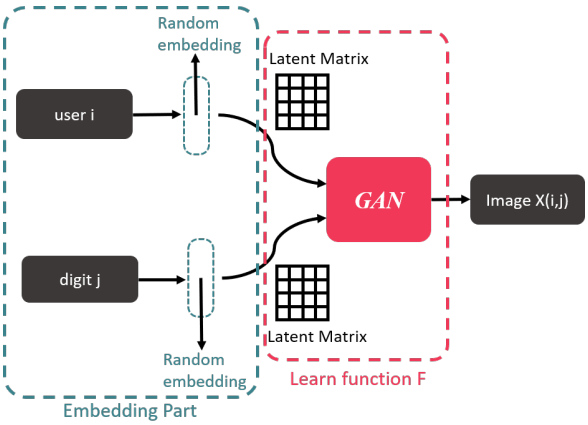

Figure 4: Diagram for Approach 1

Figure 4 shows the diagram of our first approach.

For the embedding part, we use random embedding for each user or each item.

For the part of learning the function F, we utilize GAN here. The input to the Generator would be a matrix created by concatenating user-specific sample images and digit(domain)-specific sample images. So for the FEMNIST dataset, we will have an input of size $28 \times 28 \times 2$. The generator would learn the mapping F.

The generator would be expected to generate an image which is from the domain of user and digit. We will train it using adversarial loss.

The discriminator would be expected to determine if the generated image is from the correct domain, which is the assigned $X(i, j)$.

6.2 Approach 2

Our second approach differs from the first one in their embedding part. The diagram for approach 2 is shown in Figure 5.

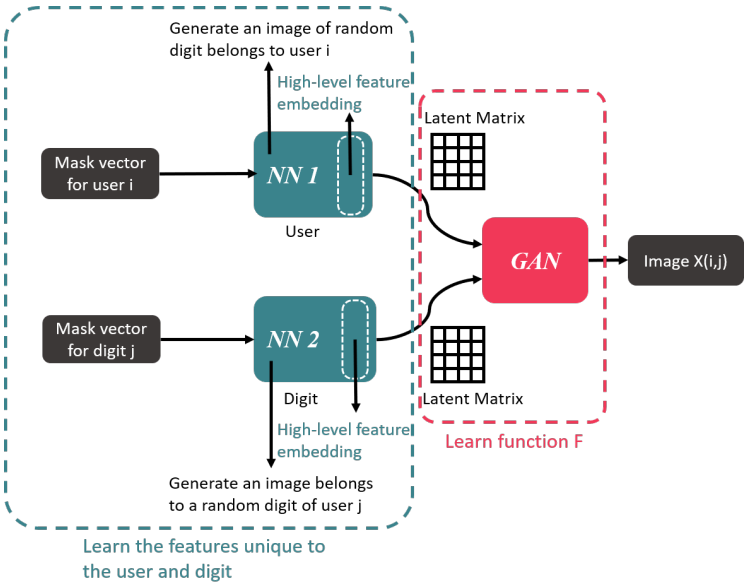

Figure 5: Diagram for Approach 2

Instead of random embedding, our second approach uses Neural Network (NN) for embedding. The loss function would be sum of correct domain and correct user. Using high level features extracted from NN trained for domain classification may yield a strong prior which will have some possibilities to improve the performance. However, this will also definitely lead to a complicated approach and we are not very confident this complicated one will bring us a better result.

Since the dataset we used have different numbers of samples for each location (i,j), Approach 1 and Approach 2 should select the dataset first so that they can have same numbers of samples. After selection, there are only about 3,500 users' less than 35,000 samples available, which will be a challenge for our training. Also, the github code for CollaGAN we planned to use is not universal, it has many parameters that only targets their dataset and their dataset is not available to us too, so we can not do test on their code first. Therefore, I finally gave up with these two approaches at this time, but will continue on them in the near future.

6.3 Approach 3

This approach is based on the similarity of different users' handwriting. We know that GAN is a method that has great performance in unsupervised machine learning, however, we should admit that clustering also performs successfully in unsupervised area. How about combine these two fantastic methods together?

First, we will do clustering on $user_i$ using Kmeans for each digit. Users with similar handwriting in one digit will be clustered together. The following is the diagram for clustering users in each domain of digits.

Digit: j	**0**	**1**	**2**	⋯	**9**
Cluster_a	User_0 User_1 User_3	User_1 User_4 User_7			
Cluster_b	User_2 User_9 User_10	User_0 User_2			User_1 User_10
Cluster_c	User_4 User_7 User_5 User_6	User_10	User_2 User_3		User_0
⋮					
Cluster_g	User_7 User_8	User_3 User_6	User_0 User_1 User_4		

Figure 6: Cluster users for each digit

To ensure that we can have enough samples to feed the GAN, having about 500 users in each cluster will be feasible. After calculation, finally, we decided to use 7 clusters for all the users in the domain of each digit. The good thing is that this time we do not need to keep the numbers of the samples for each location (i, j) the same. And we can will have about 3,500 users' 390,000 samples in total available, which will be a great preparation for utilizing GAN here to generate convincing results.

We assume $\forall x \in \mathcal{X}, \forall i \in \mathcal{I}, x \in \{x | x \in ClusterC_a\}$, $A(i, x)$ is almost the same as each other, where x is one sample image from the dataset, i is the user i, $A(i, x)$ is the possibility that $user_i$ writes the image x. This means the possibility to write the the image x is almost the same.

After clustering, we can use the results of Kmeans to build the strategy of feeding the GAN. For $user_i$, find his/her cluster for the available digit, then mark all the other users' name in this cluster of

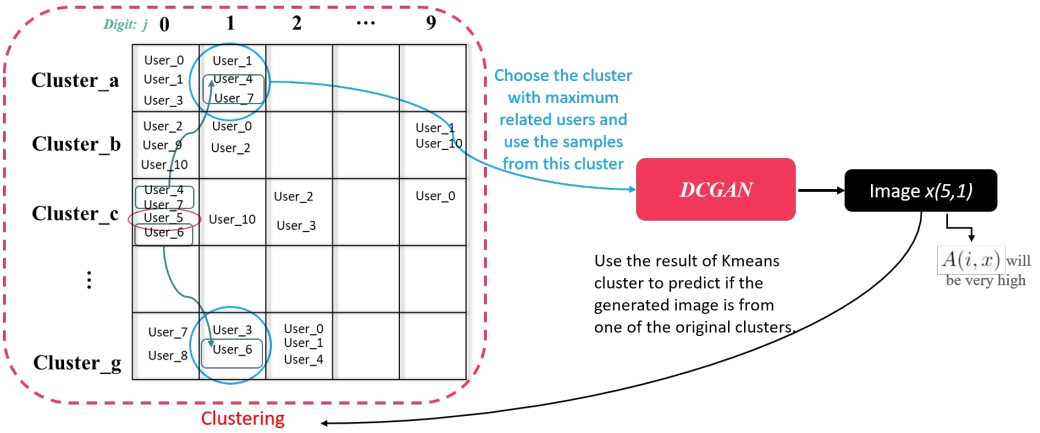

Figure 7: Diagram for Approach 3

the available digit domain. After that, we can count the numbers of these related users in each cluster in the assigned digit domain. Choosing the cluster with the maximum related users is also a very important step because when we choose all the related clusters, then there may be some cases that all the clusters are connected with our assigned location (i, j), which will make our GAN training results not targeting the assigned location. Finally, we can feed our DCGAN with the sample images from this chosen cluster in the assigned digit domain so that we can generate a new image to fill it.

GAN is to sample from the distribution of the other users' data in the cluster, so the generated image will also keep this distribution. Therefore, the $A(i, x)$ should be very high for this generated image. Therefore, we can use the results of Kmeans again to predict if the generated images is from the chosen cluster. The whole diagram for approach 3 has been shown above.

For example, if we would like to generate sample images for location $(5, 1)$, which means generate images for $user_5$ writing digit 1. We can see from the above diagram that, $user_5$'s sample is available at digit domain 0. Therefore, we can list all the users that is in the same cluster as $user_5$ in domain digit 0, which are $user_4, user_6, user_7$. When we shift our sight to the assigned digit domain 1, these related users are in different clusters. There are 2 related users in $cluster_a$ and 1 related users in $Cluster_g$. We should pick the cluster with the maximum numbers of related users which is $Cluster_a$ in this example. Then, we can use all the sample images from ($Cluster_a$,digit domain 1) to generate sample images for the assigned location $(5, 1)$. Here, the possibility for this $user_5$ to write the the generated image $x(5, 1)$, which is $A(5, x(5, 1))$, will be very high.

7 Experiment settings

7.1 Dataset used

MNIST dataset[5] consists of images of digits written by different users. If we throw away the user identity then the complete set of images is the MNIST dataset. However, if we preserve the identities then such a dataset can conform to our problem statement.

FEMNIST[2] is a dataset created by Carnegie Mellon University based on MNIST with 3550 users and is classified with both users and characters. It has 80,5263 samples in total and 226.83 samples per user, which is enough for our project.

However, if we use FEMNIST with no preprocessing, it will bring a problem of not having same number of samples for each user of every digit for Approach 1 and 2. I tried to solve this problem through manual selection for each location $X(i, j)$ in our sample matrix at the first step, however the remaining data may not be sufficient for our training. Therefore, we finally end up in diving into Approach 3, which will not be influenced by this issue. For this approach, what we should do is to pick all the digit samples out.

7.2 Experiment Environment

- **Pytorch** This widely-used optimized tensor library in machine learning is needed for our project.
- **Google Cloud Platform** Google Cloud is a platform that has great flexibility. We can build our own AI platform on it so that we can write and execute Python in our browser. GPU we used in this platform: Nvidia Tesla K80.

8 Experiment Results

8.1 Proof of cluster-GAN-cluster structure

This part of experiment is to show that when the DCGAN is feeded with the sample images from one clusters using the result of Kmeans Clustering, the generated image from DCGAN will have high possibility to still be clustered in the same cluster where the training dataset is in. The diagram for this progress is shown below.

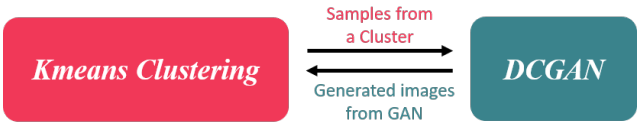

Figure 8: Cluster-GAN-Cluster Structure

When we train our DCGAN without a clustering part, the generated images will be like a random generation progress that can only be in the domain of the digit. The training and generated images are shown below.

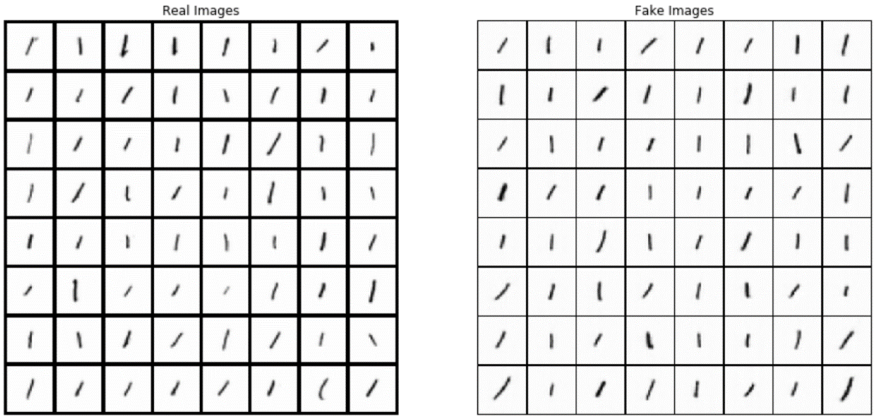

Figure 9: The training and generated images without a cluster

The loss value of the generator and the discriminator during training can been found in Figure 10.

When we add the clustering part, we can see obviously from the below images that the training dataset has different features for different clusters, the generated samples will also inherit these features from the input dataset. The training samples and generated images can be found in Figure 13, Figure 14, Figure 15, and Figure 16.

The loss value of the generator and the discriminator during training for each cluster can been found in Figure 17, Figure 18, Figure 19, and Figure 20.

More results for other digits with clustering are shown in Figure 21, Figure 22, Figure 23, and Figure 24. We can also see that they also performs great.

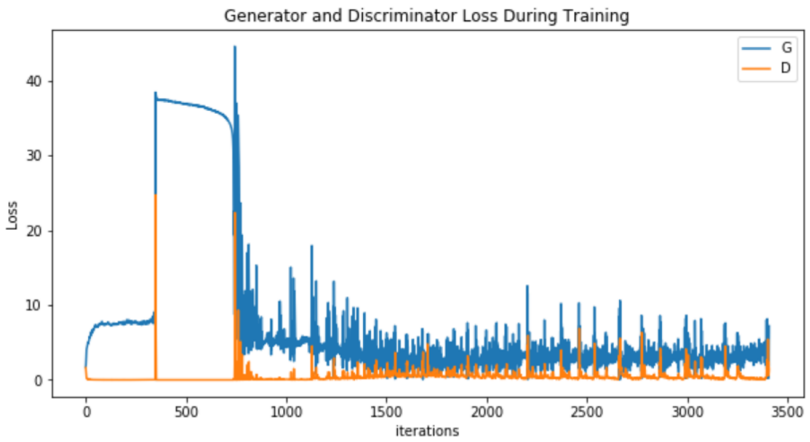

Figure 10: The loss of the generator and the discriminator during training without a cluster

We define the accuracy of the generated image in the correct cluster as: $Accuracy = \frac{N_{correct}}{N_{all}}$, where $N_{correct}$ is the number of the generated images be clustered in the correct cluster, N_{all} is the number of all the generated images. Through testing, we can get the accuracy of digit 0 as $Accuracy_{digit_0} \geq 80\%$, and the accuracy of digit 2 as $Accuracy_{digit_2} \geq 85\%$. Due to time limitation, we only do this accuracy experiment on two digits. However, these two results have already verified that the generated image from DCGAN will have high possibility to be clustered in the same cluster where the training dataset is in and our cluster-GAN-cluster structure works successfully. This is both a foundation and a feasibility proof of our Approach 3.

8.2 Assign a missing data location and generate images to fill it

After proving the feasibility of our cluster-GAN-cluster structure, we should now do the job of assign a missing data location and then generate images use Approach 3 to fill this vacant location.

Figure 11: Generated fake image for $user_{f4046\_46}$ digit 1

The real images for this location is shown as following.

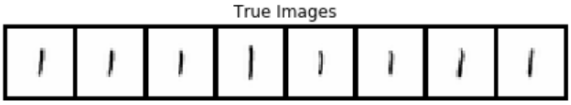

Figure 12: Real image for $user_{f4046\_46}$ digit 1

These two series of images shares similar features. More interesting is that some fake images look like the same as the true images. Therefore, it can be confirmed that more applications based on our approach of cluster-GAN-cluster can be pursued in the near future.

9 Steps to complete work

Our baseline is to evaluate the performance of CollaGAN by ourselves quantitatively and qualitatively on FEMNIST dataset. Then we will implement our own approach and do some comparisons with the baseline. For evaluation part, we are planning to do it both qualitatively and quantitatively. The

qualitative part will be conducted by showing sample outputs, and quantitative part will be done through commonly used Inception Score [6] or Frechet Inception Distance [7].

The rough timeline to complete the work:

- **6.10 - 6.17:** Finish literature review and design concrete approaches to improve the baseline. Also, prepare for the 3-min project pitch.
- **6.17 - 6.23:** Implement the baseline on the dataset of FEMNIST and evaluate the baseline. Finish the finalized proposal.
- **6.24 - 7.14:** Implement our own approach and compare it with the baseline. However, the results is not that good, so we transferred to a cluster-GAN-cluster method. Prepare for mid-point check in and 5-min updates.
- **7.15 - 7.21:** Wrap up mid-point codes and submit sample code.
- **7.22 - 8.3:** Prepare for the final presentation.
- **8.4 - 8.14:** Wrap up everything and finish the final report.

10 Summary and Conclusions

In this project, we proposed a novel cluster-GAN-cluster approach towards Personalized Data Generation, which combines the advantages of both clustering and GAN. For the clustering part, we created an interesting mapping strategy for clustering.

We utilized our approach to the dataset of FEMNIST, which brings a high-quality and convincing output. The problem of different numbers of samples for each location in the dataset has also been solved.

We have done the the qualitative evaluation by showing convincing sample outputs. We also tested the feasibility of using the method of cluster-GAN-cluster. After Kmeans Clustering, the generated image from DCGAN will have high possibility (at least 80%) to still be clustered in the same cluster where the training dataset is in.

References

[1] 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*, pages 2672–2680, 2014.

[2] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[3] Dongwook Lee, Junyoung Kim, Won-Jin Moon, and Jong Chul Ye. Collagan: Collaborative gan for missing image data imputation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2487–2496, 2019.

[4] Kai Zhong, Zhao Song, Prateek Jain, and Inderjit S. Dhillon. Nonlinear inductive matrix completion based on one-layer neural networks. *CoRR*, abs/1805.10477, 2018.

[5] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[6] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

11 Appendix

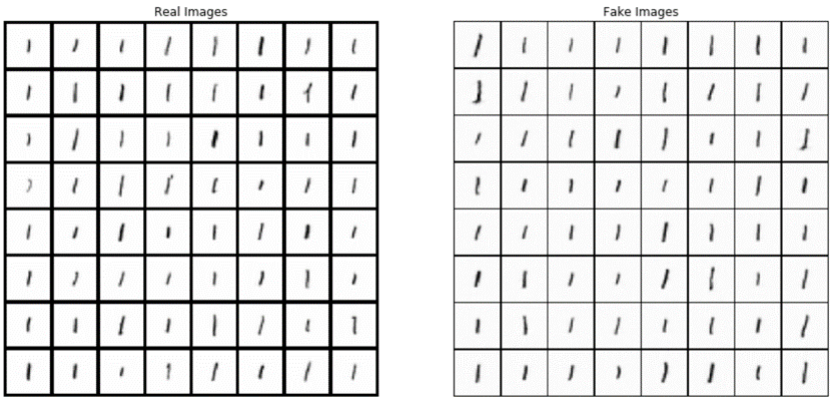

Figure 13: The training dataset and generated images of digit 1 cluster 0

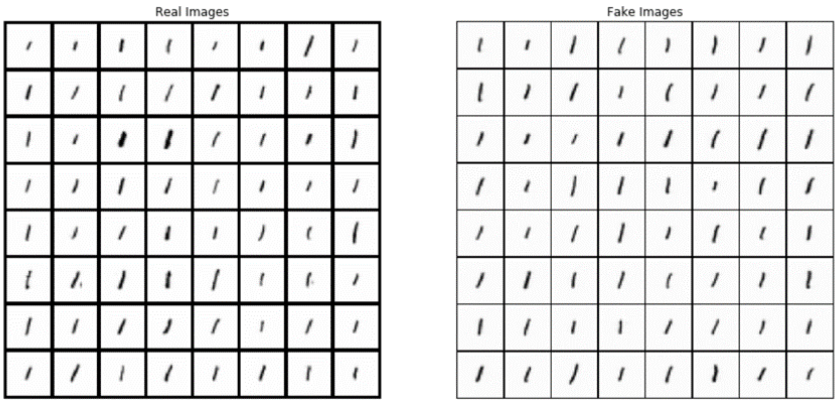

Figure 14: The training dataset and generated images of digit 1 cluster 1

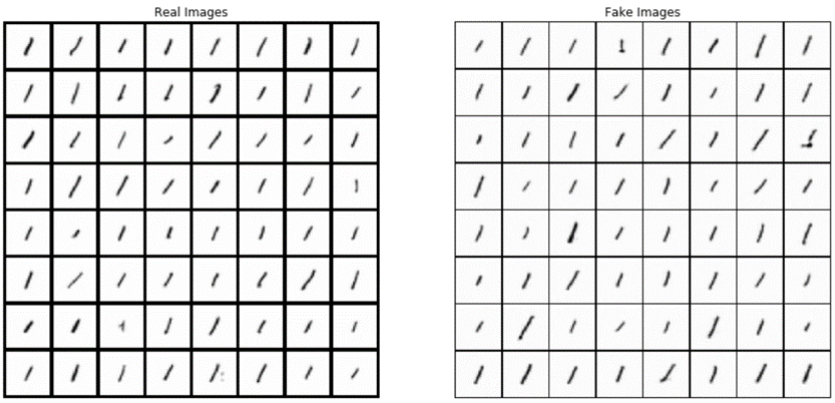

Figure 15: The training dataset and generated images of digit 1 cluster 2

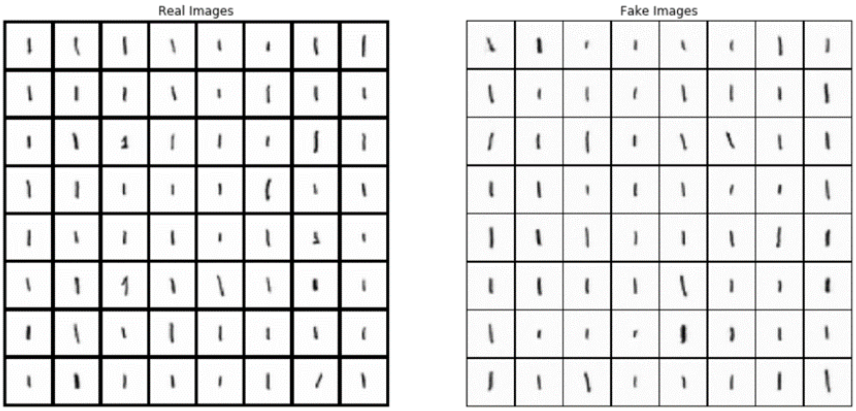

Figure 16: The training dataset and generated images of digit 1 cluster 3

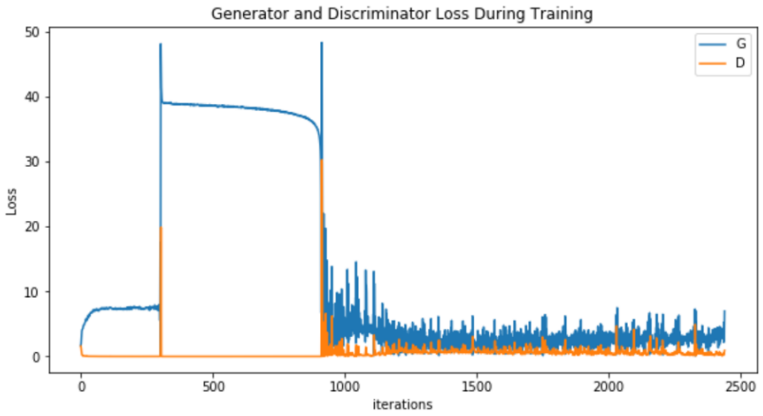

Figure 17: The loss of the generator and the discriminator during training of digit 1 cluster 0

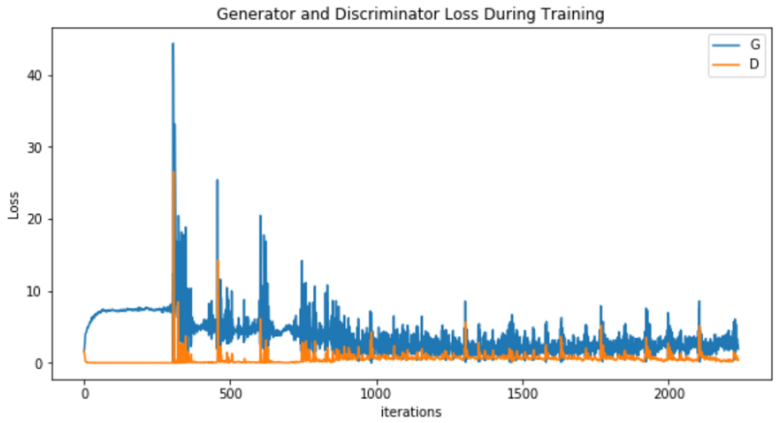

Figure 18: The loss of the generator and the discriminator during training of digit 1 cluster 1

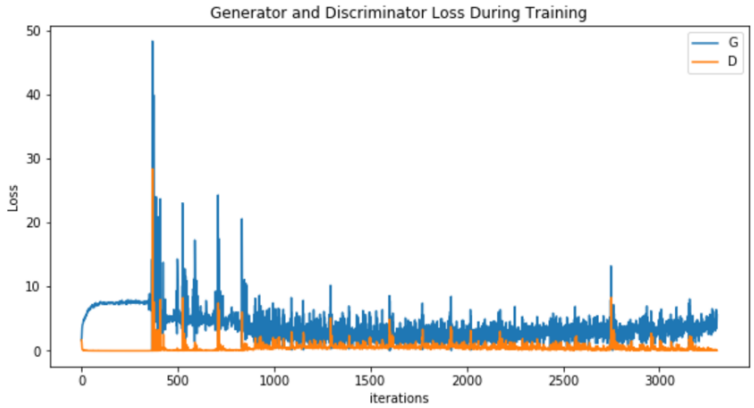

Figure 19: The loss of the generator and the discriminator during training of digit 1 cluster 2

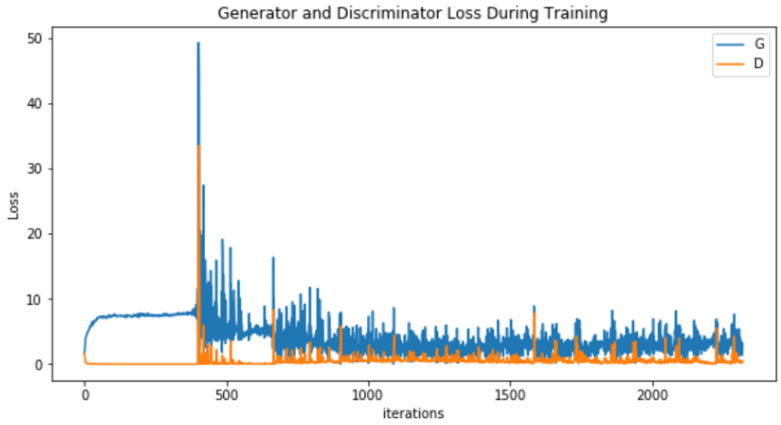

Figure 20: The loss of the generator and the discriminator during training of digit 1 cluster 3

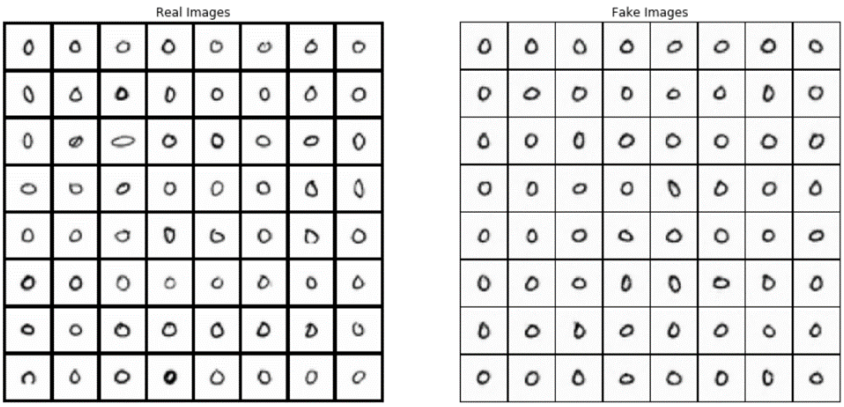

Figure 21: The training dataset and generated images of digit 0 cluster 0

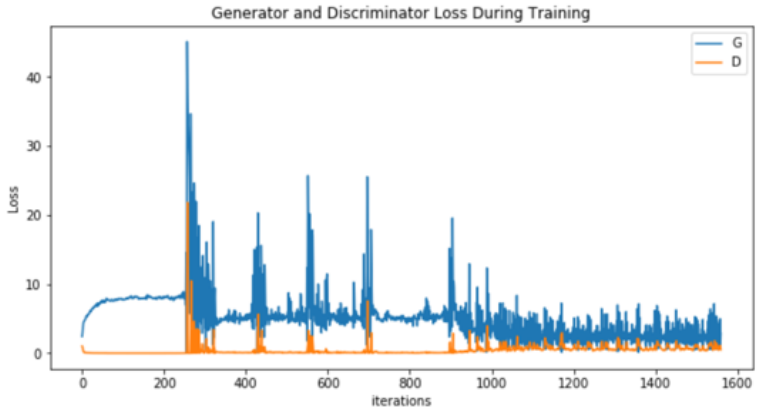

Figure 22: The loss of the generator and the discriminator during training of digit 0 cluster 0

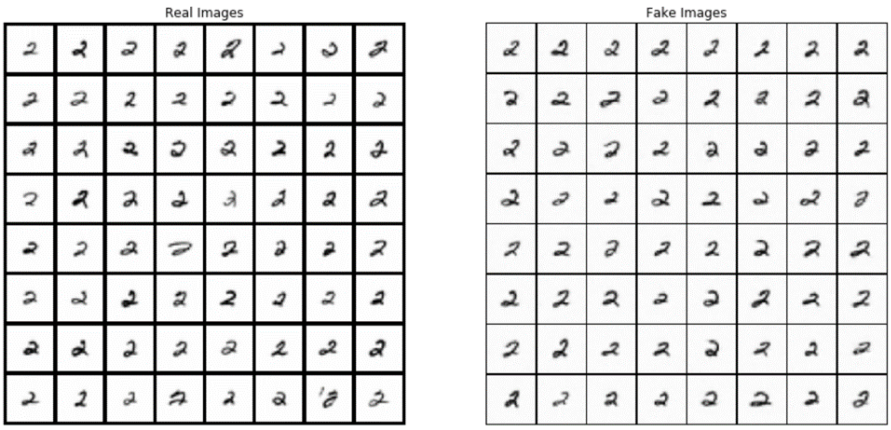

Figure 23: The training dataset and generated images of digit 2 cluster 0

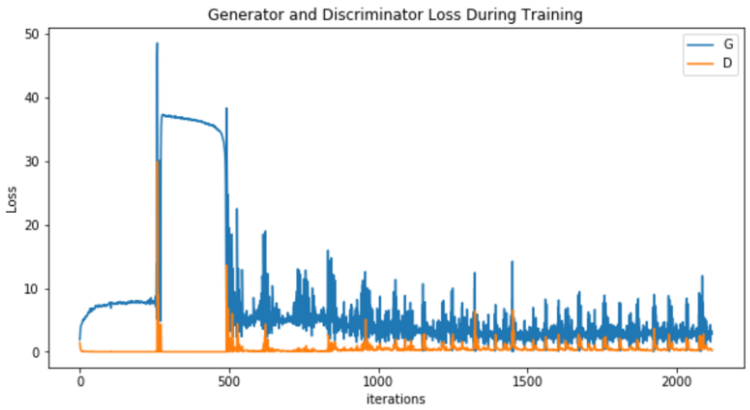

Figure 24: The loss of the generator and the discriminator during training of digit 2 cluster 0