Training self-supervised class-conditional GAN by adjusting categorical latent distribution

Jeongik Cho
jeongik.jo.01@gmail.com

Abstract

Class-conditional GAN is a conditional GAN that can generate class-conditional distribution. Among class-conditional GANs, InfoGAN with categorical latent distribution can generate class-conditional data through a self-supervised (unsupervised) method without labeled data. Instead, InfoGAN requires optimal categorical latent distribution to train the model.

In this paper, we propose a novel GAN that allows the model to perform self-supervised class-conditional data generation and clustering. The proposed method uses Bayesian inference to estimate optimal categorical latent distribution from the classifier output distribution. In the proposed method, based on the classifier output distribution of the fake data and the current categorical latent distribution, the categorical latent distribution is updated to fit the classifier output distribution of the real data. As training progresses, the entropy of the categorical latent distribution gradually decreases and converges to the appropriate value. The approximated categorical latent distribution becomes appropriate to represent the discrete part of the data distribution.

The proposed method does not require labeled data, optimal categorical latent distribution, and a good metric to calculate the distance between data. Also, a classifier used in training can be used for clustering.

1 Introduction

Class-conditional GAN is a conditional GAN [2] that can generate class-conditional distribution when a labeled dataset is given. In general, class-conditional GAN takes a latent distribution and a categorical distribution as inputs and generates class-conditional distribution. ACGAN [3] and CAGAN [4] are examples of class-conditional GANs. However, these class-conditional GANs can only be trained given labels, which is the conditional categorical distribution of the dataset. Therefore, these methods cannot be utilized for unlabeled datasets.

Unlike ACGAN or CAGAN, InfoGAN [5] with categorical latent distribution can train a class-conditional GAN even if the data is not labeled. However, InfoGAN requires optimal categorical latent distribution. It includes the number of categories and the probability for each category. For example, to perform class-conditional generation with InfoGAN on the MNIST handwritten digits dataset [9] without labels, we need to know the number of categories (10 categories) and the probability of each category (0.1 for each category) for categorical latent distribution.

In this paper, we introduce a self-supervised (unsupervised) class-conditional GAN (SCGAN) that is capable of generating class-conditional data without being given labels and optimal categorical latent distribution.

SCGAN uses only adversarial loss and classification loss, the same as InfoGAN with the categorical latent distribution. In other words, the generator takes a continuous latent distribution and categorical latent distribution as inputs and is trained so that the generated data is correctly classified by the classifier. The discriminator can share all hidden layers with the classifier, and the classifier is trained to predict the categorical latent distribution of the generated data.

The difference between InfoGAN and SCGAN is that SCGAN gradually changes the categorical latent distribution during training. Given a classifier output distribution and a categorical latent distribution, SCGAN uses Bayesian inference to predict the optimal categorical latent distribution through the classifier output distribution of real data distribution. During the training, SCGAN incrementally changes the current categorical latent distribution based on the predicted optimal categorical latent

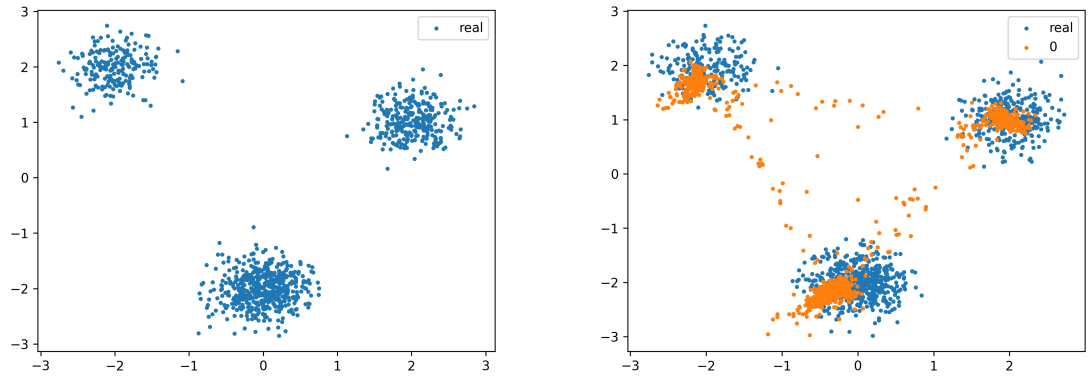

Figure 1: Left part: Two-dimensional dataset consisting of three Gaussian clusters. The centers and probabilities of each cluster are $(-2.0, 2.0), (2.0, 1.0), (0.0, -2.0)$ and $0.2, 0.3, 0.5$, respectively. The standard deviation for all clusters is 0.3. Right part: The data distribution generated by a GAN trained with a continuous latent distribution to generate the data distribution on the left part.

distribution. If the categorical latent distribution is optimal and the model is converged, SCGAN can generate class-conditional distribution and can perform clustering through the classifiers.

Therefore, unlike InfoGAN, SCGAN can be trained even if the optimal categorical latent distribution is not known. Also, same as InfoGAN, SCGAN does not require labeled data or a good metric to calculate the distance between data. Furthermore, SCGAN can perform clustering through the classifier which was used during the training.

2 Class-conditional data generation

Typically, when training a GAN, everything is assumed to be continuous. This means that the data distribution and latent distribution are assumed to be continuous, and the generator and discriminator of GAN are assumed to be continuous functions (deep learning models must be differentiable continuous functions in order to be trained).

However, the data distribution is not necessarily continuous. When data distribution includes a discrete part and latent distribution is continuous, a sufficiently complex deep generative model can approximate the discrete part of the data distribution. However, still, approximating the discrete part of the data distribution is not easy for most deep generative models, which is a continuous function.

The left part of Fig. 1 shows a data distribution example consisting of three Gaussian clusters. There is no discrete part in this data distribution, but one can see that it is easier to represent this data distribution with a discrete (categorical) latent distribution.

The right part of Fig. 1 shows generated data with GAN and continuous latent distribution. One can see that the model generates lines connecting each cluster. This is because the latent distribution is continuous and the generator is a continuous function, making it difficult to represent the discrete part of the data distribution. As training progresses, the probability density of the line connecting the clusters decreases, but it requires a long training period, and it is hard to say that the continuous latent distribution correctly represents the real data distribution.

For datasets with discrete parts, such as the example in Fig. 1, using a discrete latent distribution is more appropriate for model training and data representation. Class-conditional generative models, such as ACGAN [3] and CAGAN [4], take both continuous latent distribution and discrete categorical latent distribution as inputs and generate class-conditional distribution. It makes them appropriate for representing datasets with discrete parts. However, ACGAN and CAGAN cannot be trained if there is no label for data.

InfoGAN [5] can perform class-conditional data generation and inversion (clustering) by maximizing mutual information of generator input categorical latent distribution and classifier output distribution, even if the data is not labeled. Following equations show losses for InfoGAN with the categorical latent

distribution.

$$L_{cls} = cross\ entropy(C, Q(G(Z, C))) \tag{1}$$

$$L_d = L^d_{adv} + \lambda_{cls} L_{cls} \tag{2}$$

$$L_g = L^g_{adv} + \lambda_{cls} L_{cls} \tag{3}$$

Eq. 2 and Eq. 3 show the discriminator loss and generator loss of InfoGAN, respectively. In Eqs. 2 and 3, L^d_{adv} and L^g_{adv} represent adversarial losses [6] for GAN training. L_{cls} and λ_{cls} represent classification loss and classification loss weight, respectively. In Eq. 1, L_{cls} is cross entropy between categorical latent distribution C and predicted label of generated data $G(Z, C)$. Q and G represent the classifier and generator, respectively. Z represents continuous latent distribution. In general, a classifier Q shares all its layers with a discriminator D for efficiency.

From the above equations, one can see that a discriminator (more precisely, a classifier integrated with a discriminator) and a generator are trained to minimize classification loss. InfoGAN has shown that, given an appropriate categorical latent distribution C, it can perform class-conditional data generation and clustering (inversion) even when the data is unlabeled.

However, InfoGAN still needs insight into the categorical latent distribution C. Without knowing the appropriate categorical latent distribution C, InfoGAN still cannot perform class-conditional data generation and clustering (inversion).

In this paper, we introduce SCGAN, which performs class-conditional data generation and clustering under more general conditions than InfoGAN. We assume the following general conditions:

1. All data is unknown to which cluster it belongs (i.e., there are no labels for all data).
2. Optimal categorical latent distribution is unknown.
3. Metric to measure the distance between the data is unknown.

Under these conditions, ACGAN and CAGAN cannot be used due to condition 1. InfoGAN cannot be used due to condition 2, and recent methods utilizing the K-means algorithm cannot be used due to condition 3. On the other hand, SCGAN can still perform class-conditional data generation and clustering (inversion) even under these conditions.

3 Using Categorical latent random variable for GAN

SCGAN uses the same loss as InfoGAN with the categorical latent distribution. The difference between InfoGAN and SCGAN is that SCGAN gradually changes the categorical latent distribution during training. SCGAN uses Bayesian inference to predict the optimal categorical latent distribution. The Bayesian inference is based on the current categorical latent distribution and classifier output distribution of fake data. Given the classifier output distribution of real data, SCGAN updates the categorical latent distribution to be optimal.

Algo. 1 shows the training steps for SCGAN. The training step of SCGAN requires X (data random variable), Z (continuous latent random variable), C (categorical latent random variable), $P(C'|C)$ ($d_c \times d_c$ shape conditional probability table), D (discriminator integrated with classifier), G (generator), and k (categorical latent distribution entropy threshold).

In lines 1-3, the *sample* function represents the sampling function from a random variable. x (real data point), z (continuous latent code), and c_f (fake categorical latent code) are sampled from X, Z, and C, respectively.

In line 4, G generates fake data x' with z and c_f. In line 5, D takes a fake data point x' as input and outputs two values a_f (fake adversarial value) and c'_f (fake categorical latent code prediction). Since the discriminator and classifier are integrated, the discriminator D outputs a 1-dimensional adversarial value and d_c-dimensional categorical latent code prediction. a_f and c'_f represent fake adversarial value and predicted label of c_f, respectively Similarly, on line 6, D takes a real data point x as input and outputs adversarial value a_r and real data categorical latent code prediction c'.

In line 7, L_{cls} represents classification loss. *crossentropy* is a function that calculates cross-entropy loss. In lines 8 and 9, L_d and L_g represent discriminator loss and generator loss, respectively. f_d and f_g represent adversarial functions for GAN. λ_{cls} is classification loss weight.

Algorithm 1 Training step of SCGAN

Require: $X, Z, C, P(C'|C), D, G, k$

1: $x \leftarrow sample(X)$
2: $z \leftarrow sample(Z)$
3: $c_f \leftarrow sample(C)$

4: $x' \leftarrow G(z, c_f)$
5: $a_f, c'_f \leftarrow D(x')$
6: $a_r, c' \leftarrow D(x)$

7: $L_{cls} \leftarrow cross\ entropy(c_f, c'_f)$
8: $L_d \leftarrow f_d(a_r, a_f) + \lambda_{cls} L_{cls}$
9: $L_g \leftarrow f_g(a_f) + \lambda_{cls} L_{cls}$

10: $c \leftarrow \Sigma_{i=1}^{d_c} c'_i \cdot P(C|C' = c_i)$
11: **if** $k < \Sigma_{i=1}^{d_c} - c_i \cdot \log c_i$ **then**
12: $\quad P(C) \leftarrow update(P(C), c)$
13: **else**
14: $\quad P(C) \leftarrow update(P(C), [1/d_c, 1/d_c, ..., 1/d_c])$
15: **end if**

16: $P(C'|C = c_f) \leftarrow update(P(C'|C = c_f), c'_f)$

17: $return\ L_d, L_g, C, P(C'|C)$

In line 10, c is expected probability vector of $P(C|C' = c')$. c'_i represents i-th element of real data categorical latent code prediction c'. d_c represents a dimension of categorical latent distribution. Since $P(C'|C)$ ($d_c \times d_c$ shape conditional probability table) and $P(C)$ are already given, we can calculate expectation of $P(C|C' = c'_r)$. $P(C)$ is initialized to have maximum entropy ($[1/d_c, 1/d_c, ..., 1/d_c]$), then gradually decrease with updates. In line 11, compare entropy threshold k and entropy of c (c_i represents i-th element of c). The entropy threshold k serves to determine the minimum entropy of the categorical latent distribution. In line 12, if the entropy of c is greater than k, update the categorical probability of C with c. The *update* function can be a simple moving average, an exponential moving average, or others. If the entropy of c is not greater than k, this means that the entropy of C can be less than k after the update. To avoid this, in line 14, update the categorical probability of C with $[1/d_c, 1/d_c, ..., 1/d_c]$, which has maximum entropy, instead of c.

In line 16, $P(C'|C)$, which is $d_c \times d_c$ shape conditional probability table, is updated with c'_f Specifically, the row corresponding to class c_f in the $d_c \times d_c$ shape matrix is updated to be closer to c'_f. As before, the *update* function can be a simple moving average, an exponential moving average, etc.

In Algo. 1, one can see that SCGAN uses the same loss as InfoGAN. The difference between SCGAN and InfoGAN is that in SCGAN, $P(C)$ changes as the training progresses. The entropy of the categorical latent distribution C converges to an appropriate value as the model is trained. The conditional probability table $P(C'|C)$ is used to compute $P(C|C')$ in the process of updating $P(C)$.

4 Experiments

We trained the models to generate two-dimensional Gaussian clusters distribution and the MNIST dataset [9]. In Gaussian clusters experiments, we compare the performance of Vanilla GAN [1], InfoGAN [5], and our proposed SCGAN. In MNIST experiments, we compared the clustering of SCGANs according to an entropy threshold k.

The following hyperparameters were used for experiments.

$$\lambda_{r1} = 1.0$$
$$Z \sim N(0, I_{d_z})$$

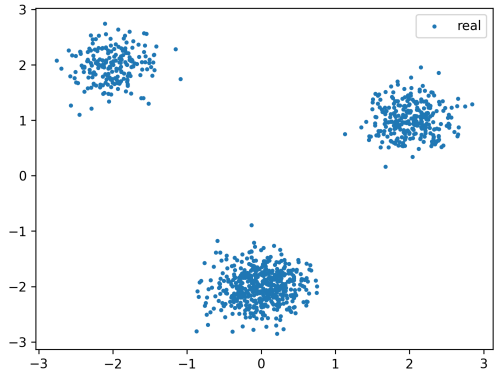

Figure 2: A two-dimensional dataset consisting of three Gaussian clusters. The centers and probabilities of each cluster are $(-2.0, 2.0), (2.0, 1.0), (0.0, -2.0)$ and $0.2, 0.3, 0.5$, respectively. The standard deviation for all clusters is 0.3.

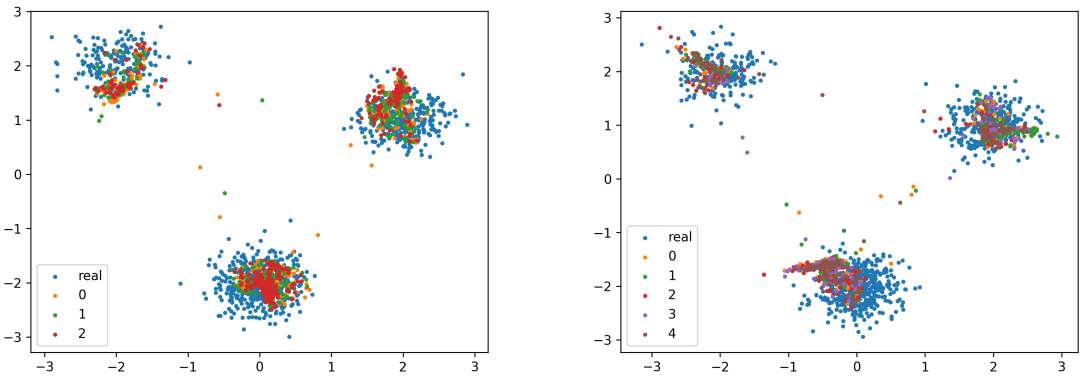

Figure 3: Vanilla GAN (trained only with adversarial loss) training with the categorical latent distribution. Left: 3-dimensional categorical latent distribution. Right: 5-dimensional categorical latent distribution. Each category has the same probability.

$$optimizer = Adam \begin{pmatrix} learning\,rate = 0.001 \\ \beta_1 = 0.0 \\ \beta_2 = 0.99 \end{pmatrix}$$
$$batch\,size = 32$$
$$train\,step\,per\,epoch = 2000$$

λ_{r1} represents R1 regularization [7] loss weight. Classification loss weight $\lambda_{cls} = 1.0$ was used for InfoGAN and SCGAN. We used exponential moving average with $decay\,rate = 0.999$ as $update$ function for SCGAN. Equalized learning rate [8] was used for all weights.

4.1 Gaussian clusters experiments

In this experiment, we used the dataset consisting of three 2-dimensioanl Gaussian clusters as a training dataset. A generator and discriminator consisting of four fully connected layers with 512 units were used for training. We used $epoch = 50$ and $d_z = 8$ (continuous latent distribution dimension) for model training. Entropy threshold k for SCGAN was not used in this experiment (i.e., $k = 0$).

Fig. 2 shows training data distribution. One can see that there are three Gaussian clusters in data distribution.

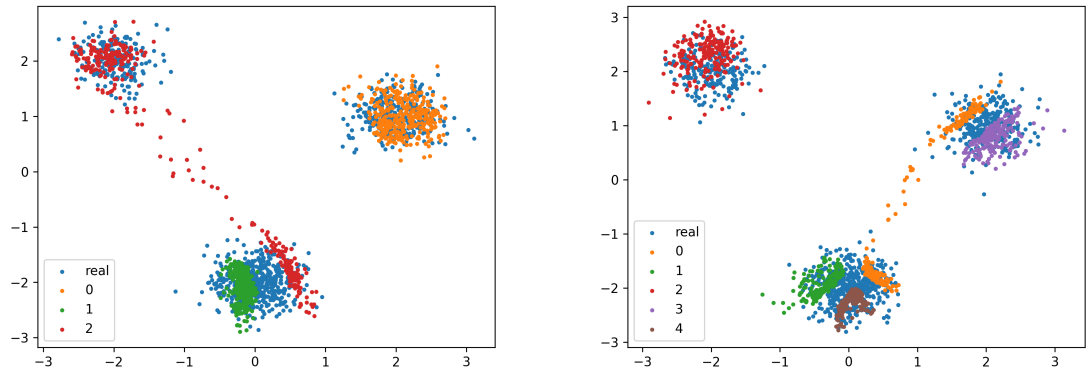

Figure 4: InfoGAN training with the categorical latent distribution. Left: 3-dimensional categorical latent distribution. Right: 5-dimensional categorical latent distribution. Each category has the same probability.

Fig. 3 shows samples generated with vanilla GAN (trained only with adversarial loss). GAN in the left part of Fig. 3 used d_z-dimensional i.i.d. normal latent distribution and 3-dimensional categorical distribution for training. The right part used a 5-dimensional categorical distribution. Each category in a categorical distribution has the same probability.

One can see that a categorical latent distribution that does not maximize mutual information via a classifier was not meaningful in data generation. In addition, since Vanilla GAN did not meaningfully use discrete categorical latent distributions, training was practically exclusively performed by continuous latent distributions. Therefore the output space was also continuous, which caused a line generation between each cluster.

Fig. 4 shows samples generated with InfoGAN. Unlike the Vanilla GAN, one can see that the model generates class-conditional distribution with the categorical latent distribution. However, the class-conditional distribution was not correct. First, in the left part of Fig. 4 (3-dimensional categorical latent distribution), one can see that category 2 is split between a cluster centered at (-2, 2) and a cluster centered at (0, -2). For a cluster with a center of (-2, 2), the probability is 0.2, which is less than the probability of category 2, which is 0.333. Thus, the left portion of category 2 is also distributed in the cluster with center (0, -2) with a probability of 0.5. Also, in the cluster with the center at (0, -2), one can see that no data was generated between category 1 and category 2. This is because the model was trained to correctly classify category 1 and category 2 within the same cluster. Furthermore, since there are no other discrete latent distributions within the same category, one can see line generation like a vanilla GAN. Similar problems are occurring in the right part of Fig. 4 (5-dimensional categorical latent distribution).

Fig. 5 shows samples generated with SCGAN. Unlike InfoGAN, one can see that SCGAN creates three Gaussian clusters almost perfectly. There are no lines between clusters like in Vanilla GAN or InfoGAN, and the probability of each cluster is almost the same as in the original dataset. In particular, for the right part in Fig. 5, which used a 5-dimensional categorical latent distribution, the probability of the two unnecessary categories became very close to zero, so they did not participate in the training. This shows that SCGAN can perform correct class-conditional data generation and clustering without knowing the optimal categorical latent distribution.

4.2 MNIST experiments

In this section, we trained SCGAN to generate the MNIST handwritten digits dataset. $d_z = 64$, $d_c = 17$, $epoch = 100$ were used for the experiments.

Figs. 6, 7, and 8 show the difference in clustering of SCGAN according to the entropy threshold k. If there are 10 categories, each with a probability of 0.1, the entropy is about 2.3. Thus, the closer the entropy of the categorical distribution of SCGAN is to 2.3, the more likely it is that each category will have only one kind of number.

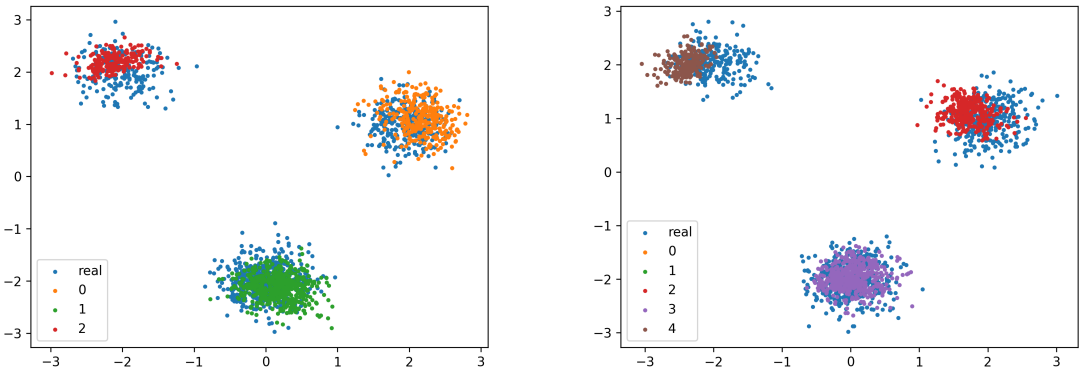

Figure 5: SCGAN training with the categorical latent distribution. Left: 3-dimensional categorical latent distribution. $P(C) = [0.2975, 0.5007, 0.2018]$ Right: 5-dimensional categorical latent distribution. $P(C) = [0.0000, 0.0000, 0.3000, 0.5022, 0.1978]$

Fig. 6 shows the clustering of SCGAN when $k = 0$. The entropy of the categorical latent distribution was 1.2671, which is smaller than 2.3. This means that several different numbers can be clustered into the same category. In Fig. 6, the first category has a probability of about 0.5. One can see that numbers $0, 2, 3, 5, 8$ were classified as the first category. The second category has a probability of about 0.2. Numbers $7, 9$ were classified as the second category. This means that the model has determined that the numbers 7 and 9 are closer than the other numbers. The third category has a probability of about 0.1. Only the number 4 was classified as the third category. The fourth category has a probability of about 0.2. Numbers $1, 6$ were classified as the third category.

If this clustering is not useful, one can increase the entropy threshold k to make SCGAN perform more detailed clustering. In Fig. 7, since the entropy threshold k is larger than in Fig. 6, each category was further refined. Except for the third category, which contains numbers $5, 8$, the rest of the categories are all single digits, and one can see that the probability of each category is about 0.1.

By further increasing the entropy threshold k, each category was further refined. In Fig. 8, some numbers were divided into multiple categories. For example, the number 3 is divided into the first category and the 12-th category. One can also see that numbers other than 3 are also divided into different categories.

5 Conclusion

In this paper, we introduced SCGAN, a self-supervised class-conditional GAN. SCGAN uses Bayesian inference to estimate optimal categorical latent distribution from the classifier output distribution. The entropy of the categorical latent distribution gradually decreases until SCGAN becomes stable. SCGAN does not require a label of data, optimal categorical latent distribution, and a good metric to calculate the distance between data. This means that SCGAN can be used in most situations regardless of the data domain. Also, SCGAN can perform clustering through the classifier used during training.

SCGAN showed the best performance compared to Vanilla GAN or InfoGAN with categorical latent distribution in Gaussian clusters generation experiments. SCGAN was also able to perform self-supervised class-conditional data generation on the MNIST experiment, a slightly more complex dataset.

References

[1] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Nets. In Commun. ACM, vol. 63, no. 11, pp. 139-144, Nov. 2020. https://doi.org/10.1145/3422622

Figure 6: MNIST generated data with entropy threshold $k = 0$. Each row has the same continuous latent code, and each column has the same categorical latent code. Out of d_c categories, those with a probability less than 1% were excluded. The probabilities for each category are $[0.4951, 0.1984, 0.0975, 0.1991]$. The entropy of categorical latent distribution is 1.2671.

Figure 7: MNIST generated data with entropy threshold $k = 1.8$. Each row has the same continuous latent code, and each column has the same categorical latent code. Out of d_c categories, those with a probability less than 1% were excluded. The probabilities for each category are $[0.0996, 0.0960, 0.1895, 0.0922, 0.1069, 0.1089, 0.0999, 0.1004, 0.0999]$. The entropy of categorical latent distribution is 2.2090.

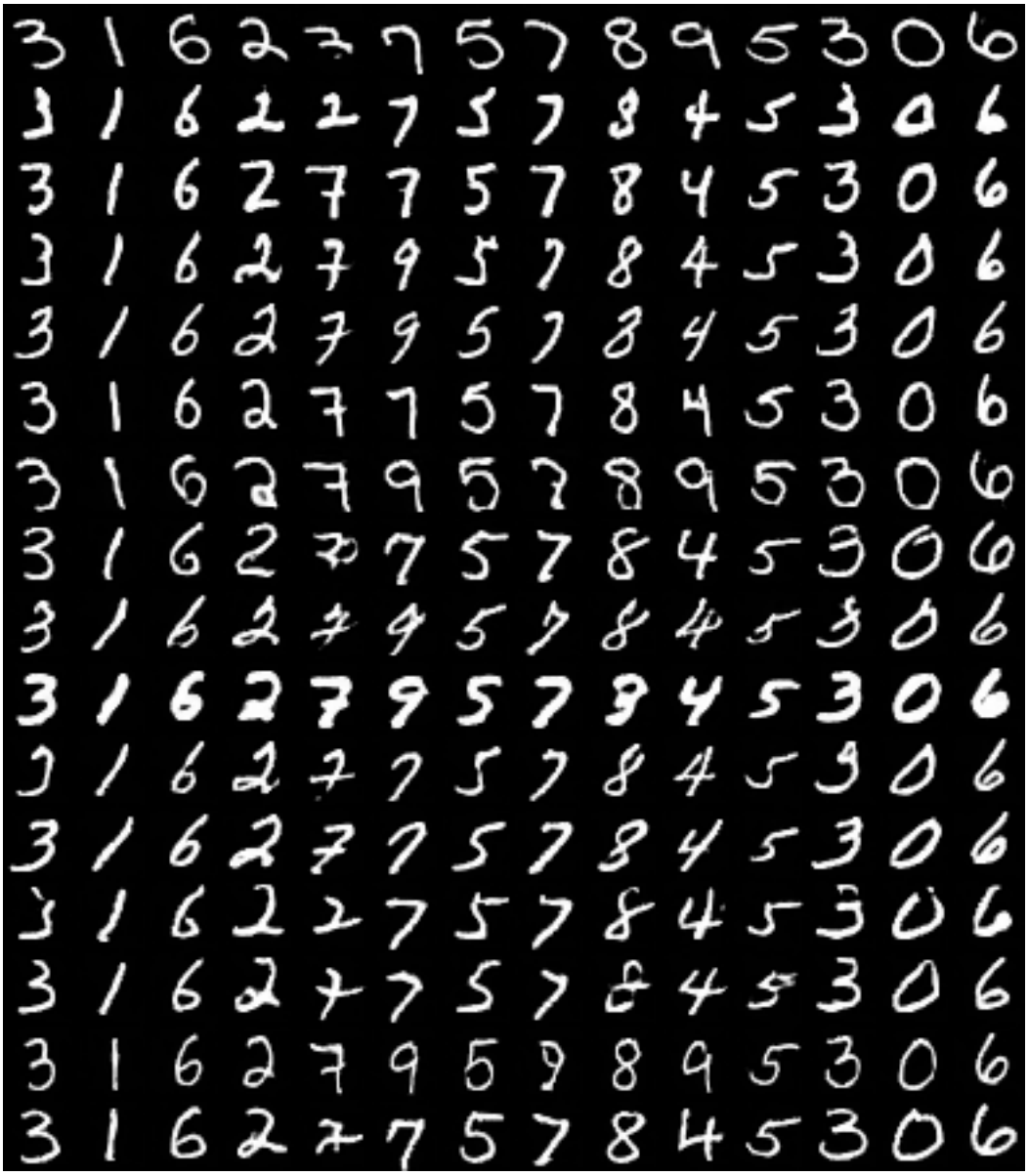

Figure 8: MNIST generated data with entropy threshold $k = 2$. Each row has the same continuous latent code, and each column has the same categorical latent code. Out of d_c categories, those with a probability less than 1% were excluded. The probabilities for each category are $[0.0638, 0.1090, 0.0262, 0.0931, 0.0205, 0.1456, 0.0687, 0.0178, 0.1190, 0.1162, 0.0196, 0.0120, 0.0993, 0.0742]$. The entropy of categorical latent distribution is 2.4848.

[2] Mirza, M., Osindero, S.: Conditional Generative Adversarial Nets. In arXiv preprint, 2014, arXiv:1411.1784. https://arxiv.org/abs/1411.1784

[3] A. Odena, C. Olah, J. Shlens, "Conditional Image Synthesis with Auxiliary Classifier GANs," in proceedings of the 34th International Conference on Machine Learning, PMLR 70:2642-2651, 2017. https://proceedings.mlr.press/v70/odena17a.html

[4] Cho, J., Yoon, K.: Conditional Activation GAN: Improved Auxiliary Classifier GAN. In IEEE Access, vol. 8, pp. 216729-216740, 2020. https://doi.org/10.1109/ACCESS.2020.3041480

[5] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. In NIPS proceedings, 2016. https://papers.nips.cc/paper/2016/hash/7c9d0b1f96aebd7b5eca8c3edaa19ebb-Abstract.html

[6] Lucic, M., Kurach, K., Michalski, M., Gelly, S., Bousquet, O.: Are GANs Created Equal? A Large-Scale Study. In NIPS, 2018. https://papers.nips.cc/paper/2018/hash/e46de7e1bcaaced9a54f1e9d0d2f800d-Abstract.html

[7] Mescheder, L., Geiger, A., Nowozin S.: Which Training Methods for GANs do actually Converge? In PMLR, 2018. https://proceedings.mlr.press/v80/mescheder18a.html

[8] Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive Growing of GANs for Improved Quality, Stability, and Variation. In ICLR conference, Vancouver, Canada, Apr. 30-May 3, 2018. https://openreview.net/forum?id=Hk99zCeAb

[9] Lecun, Y., Cortes, C., and Burges, C.: MNIST handwritten digit database, 2010. In ATT Labs. http://yann.lecun.com/exdb/mnist