Unsupervised Image Generation with Infinite Generative Adversarial Networks

Hui Ying[1],   He Wang[2],   Tianjia Shao[1*],   Yin Yang[3],   Kun Zhou[1]

[1]Zhejiang University   [2]University of Leeds   [3]Clemson University

huiying@zju.edu.cn, H.E.Wang@leeds.ac.uk, tjshao@zju.edu.cn, yin5@clemson.edu, kunzhou@acm.org

Abstract

Image generation has been heavily investigated in computer vision, where one core research challenge is to generate images from arbitrarily complex distributions with little supervision. Generative Adversarial Networks (GANs) as an implicit approach have achieved great successes in this direction and therefore been employed widely. However, GANs are known to suffer from issues such as mode collapse, non-structured latent space, being unable to compute likelihoods, etc. In this paper, we propose a new unsupervised non-parametric method named mixture of infinite conditional GANs or MIC-GANs, to tackle several GAN issues together, aiming for image generation with parsimonious prior knowledge. Through comprehensive evaluations across different datasets, we show that MIC-GANs are effective in structuring the latent space and avoiding mode collapse, and outperform state-of-the-art methods. MIC-GANs are adaptive, versatile, and robust. They offer a promising solution to several well-known GAN issues. Code available: github.com/yinghdb/MICGANs.

1. Introduction

GANs have achieved great successes in a fast-growing number of applications [19]. The success lies in their ability to capture complex data distributions in an unsupervised, non-parametric and implicit manner [13]. Yet, such ability comes with limitations, such as mode collapse. Despite a range of methods attempting to address these issues, they are still open. This motivates our research aiming to mitigate several limitations collectively including mode collapse, unstructured latent space, and being unable to compute likelihoods, which we hope will facilitate follow-up GAN research and broaden their downstream applications.

GANs normally consist of two functions: a generator and a discriminator. In image generation, the discriminator distinguishes between real and generated images, while the generator aims to fool the discriminator by generating

images that are similar to real data. The widely known mode collapse issue refers to the generator's tendency to only generate similar data which aggregate around one or few modes in a multi-modal data distribution, e.g., only generating cat images in a cat/dog dataset. There has been active research in *distribution matching* to solve/mitigate mode collapse [31, 45, 51, 56], which essentially explicitly/implicitly minimizes the distributional mismatch between the generated and real data. In parallel, it is found that latent space structuring can also help, e.g. by introducing conditions [39], noises [23], latent variables [5] or latent structures [15]. In comparison, latent space structuring does enable more downstream applications such as controlled image generation, but they normally require strong prior knowledge of the data/latent space structure, such as class labels or the cluster number in the data or the mode number in the latent space. In other words, they are either supervised, or unsupervised but parametric and prescribed.

We simultaneously tackle the latent space structure and mode collapse by proposing a new, *unsupervised* and *non-parametric* method, mixture of infinite conditional GANs or MIC-GANs. Without loss of generality, we assume an image dataset contains multiple (unlabelled) clusters of images, with each cluster naturally forming one mode. Instead of making a GAN avoid mode collapse, we make use of it, i.e. exploiting GAN's mode collapse property, to let one GAN cover one mode so that we can use multiple GANs to capture all modes. Next, doing so naturally brings the question of how many GANs are needed. Instead of relying on the prior knowledge [3, 15], we aim to learn the number of GANs needed from the data. In other words, MIC-GANs model the distribution of an infinite number of GANs. Meanwhile, we also construct a latent space according to the data space by letting each GAN learn to map one latent mode to one data mode. Since there can be an infinite number of modes in the data space, there are also the same number of modes in the latent space, each associated with one GAN. The latent space is then represented by a *convex combination* of GANs and is therefore structured.

To model a distribution of GANs, our first technical novelty is a new Bayesian treatment on GANs, with a family

---

arXiv:2108.07975v1 [cs.CV] 18 Aug 2021

of non-parametric priors on GAN parameters. Specifically, we assume an infinite number of GANs in our reservoir, so that for each image, there is an optimal GAN to generate it. This is realized by imposing a Dirichlet Process [11] over the GAN parameters, which partitions the probabilistic space of GAN parameters into a countably infinite set where each element corresponds to one GAN. The image generation process is then divided into two steps: first choose the most appropriate GAN for an image and then generate the image using the chosen GAN.

Our second technical novelty is a new hybrid inference scheme. Training MIC-GANs is challenging due to the infinity nature of DP. Not only do we need to estimate how many GANs are needed, we also need to compute their parameters. Some specific challenges include: 1) unable to compute likelihoods from GANs (a fundamental flaw of GANs) [9]; 2) lack of an explicit form of GAN distributions; 3) prohibitive computation for estimating a potentially infinite number of GANs. These challenges are beyond the capacity of existing methods. We therefore propose a new hybrid inference scheme called Adversarial Chinese Restaurant Process.

MIC-GANs are unsupervised and non-parametric. They automatically learn the latent modes and map each of them to one data mode through one GAN. MIC-GANs not only avoid mode collapse, but also enable controlled image generation, interpolation among latent modes, and a systematic exploration of the entire latent space. Through extensive evaluation and comparisons, we show the superior performance of MIC-GANs in data clustering and generation.

2. Related Work

Mode Collapse in GANs GANs often suffer from mode collapse, where the generator learns to generate samples from only a few modes of the true distribution while missing many other modes. To alleviate this problem, researchers have proposed a variety of methods such as incorporating the minibatch statistics into the discriminator [54], adding regularization [4, 60], unrolling the optimization of the discriminator [38], combining a Variational Autoencoder (VAE) with GANs using variational inference [51], using multiple discriminators [7], employing the Gaussian mixture as a likelihood function over discriminator embeddings [10], and applying improved divergence metrics in the loss of discriminator [2, 14, 37, 40, 45]. Other methods focus on minimizing the distributional mismatch between the generated and real data. For example, VEEGAN [56] introduces an additional reconstructor network to enforce the bijection mapping between the true data distribution and Gaussian random noise. MMD GAN [31] is proposed to align the moments of two distributions with generative neural networks. Most of existing methods essentially map one distribution (often Gaussian or uniform) to a data distribution with an arbitrary number of modes. This is an extremely challenging mapping to learn, leading to many issues such as convergence and inablility to learn complex distributions [46]. Rather than avoiding mode collapse, we *exploit* it by letting one GAN learn one mode in the data distribution (assuming one GAN can learn one mode), so that we can use multiple GANs to capture all modes in the data distribution. This not only naturally avoids mode collapse but leads to more structured latent space representations.

Latent Space Structure in GANs Early GANs focus on mapping a whole distribution (e.g. uniform or Gaussian) to a data distribution. Since then, many efforts have been made to structure the latent space in GANs, so that the generation is controllable and the semantics can be learned. Common strategies involve introducing conditions [36, 39, 41, 47, 59], latent variables [5], multiple generators [12, 32], noises [23, 24] and clustering [42]. Recent approaches also employ mixture of models (e.g. Gaussian mixture models) to explicitly parameterize the latent space [3, 15]. However, these methods usually require strong prior knowledge, e.g. class labels, the cluster number in the data distribution and the mode number in the latent space, with prescribed models to achieve the best performance. In this paper, we relax the requirement of any prior knowledge of the latent/data space. Specifically, MIC-GANs are designed to learn the latent modes and the data modes simultaneously and automatically. This is realized by actively constructing latent modes while establishing a one-to-one mapping between latent modes and data modes, where each GAN learns one mapping. Consequently, the latent space is structured by a convex combination of GANs. DMGAN [25] is the most similar work to ours, which employs multiple generators to learn distributions with disconnected support without prior knowledge. In contrast, MIC-GANs neither impose any assumption on the connectivity of the support, nor require multiple generators. Besides, MIC-GANs have a strong clustering capability for learning the latent modes. An alternative approach is to use Variational Autoencoder (VAE) which can structure the latent space (i.e., a single Gaussian or mixed Gaussians) during learning [8, 20, 21, 26, 43, 57], but they often fail to generate images with sharp details. We therefore focus on GANs.

3. Methodology

3.1. Preliminary

Given image data X, a GAN can be seen as two distributions $G(X|\theta^g, Z)$ and $D([0,1]|\theta^d, X)$, with $\theta = [\theta^g, \theta^d]$ being the network weights and Z being drawn from a distribution, e.g. Gaussian. θ uniquely defines a GAN. Unlike traditional GANs, we use a Bayesian approach and treat θ as random variables which conform to some prior distribution parameterized by $\Phi = [\Phi^g, \Phi^d]$. The inference of θ can

be conducted by iteratively sampling [52]:

$$p(\theta^g|Z,\theta^d) \propto (\prod_{i=1}^{N^g} D(G(z_{(i)};\theta^g);\theta^d))p(\theta^g|\Phi^g) \quad (1)$$

$$p(\theta^d|Z,X,\theta^g) \propto \prod_{i=1}^{N^d} D(x_{(i)};\theta^d) \times$$

$$\prod_{i=1}^{N^g}(1 - D(G(z_{(i)};\theta^g);\theta^d)) \times p(\theta^d|\Phi^d) \quad (2)$$

where N^g and N^d are the total numbers of generated and real images, $p(\theta^g|\Phi^g)$ and $p(\theta^d|\Phi^d)$ are *prior* distributions of the network weights and sometimes combined as $p(\theta|\Phi)$ for simplicity. For our goals, the choice of the prior is based on the following consideration. First, if X has K modes corresponding to K clusters, we aim to learn K mappings through K distinctive GANs, and each GAN is only responsible for generating one cluster of images. This dictates that the draws from the prior needs to be discrete. Second, since the K value is unknown *a priori*, we need to assume that $K \to \infty$. Therefore, we employ a Dirichlet Process (DP) as the prior $p(\theta|\Phi)$ for θs.

A $DP(\alpha,\Phi)$ is a distribution of probabilistic distributions, where α is called *concentration* and Φ is the *base distribution*. It describes a 'rich get richer' sampling [44]:

$$\theta_i|\theta_{-\mathbf{i}},\alpha,\Phi \sim \sum_{l=1}^{i-1} \frac{1}{i-1+\alpha}\delta_{\theta_l} + \frac{\alpha}{i-1+\alpha}\Phi \quad (3)$$

where an infinite sequence of random variables θs are i.i.d. according to Φ. $\theta_{-\mathbf{i}} = \{\theta_1,\ldots,\theta_{i-1}\}$. δ_{θ_l} is a delta function at a previously drawn sample θ_l. When a new θ_i is drawn, either a previously drawn sample is drawn again (with a probability proportional to $\frac{1}{i-1+\alpha}$), or a new sample is drawn (with a probability proportional to $\frac{\alpha}{i-1+\alpha}$). Assuming each θ has a value ϕ, there can be multiple θs having the same value ϕ_k. So there are only K distinctive values in a total of i samples drawn so far in Equation 3 where $K < i$. An intuitive (but not rigorous) analogy is rolling a dice multiple times. Each time one side (a sample) is chosen but overall there are only $K = 6$ possible values.

To see the 'rich get richer' property, the more ϕ has been drawn before, the more likely it will be drawn again. This property is highlighted by another equivalent representation called Chinese Restaurant Processes (CRP) [1], where the number of times the kth ($k \in K$) value ϕ_k has been drawn is associated with its probability of being drawn again:

$$\theta_i|\theta_{-\mathbf{i}},\alpha,\Phi \sim \sum_{k=1}^{K} \frac{N_k}{i-1+\alpha}\delta_{\phi_k} + \frac{\alpha}{i-1+\alpha}\Phi \quad (4)$$

where δ_{ϕ_k} is a delta function at ϕ_k, and N_k is how many times that ϕ_k has been sampled so far. Equation 3 and 4

are equivalent with the former represented by draws and the latter by actual values.

3.2. Mixture of Infinite GANs

We propose a new Bayesian GAN which is a mixture of infinite GANs model. Following Eq. 4, ϕ represents the network weights of a GAN. Imagine we have $K \to \infty$ GANs and examine X one by one. For each image x_i, we sample the best GAN ϕ_{c_i} (based on some criteria) to generate it. So $N_k = \sum \mathbf{1}_{c_i=k}$ is the total number of images already selecting the kth GAN. The more frequently a GAN is selected, the more likely it will be selected in future. If all N_ks are small, then a new GAN is likely to be sampled based on Φ. We describe the generative process of our model as:

sample $z_i \sim Z, \{\phi_1,\ldots,\phi_k,\ldots,\phi_K\} \sim \Phi$

sample $c_i \sim CRP(\alpha,\Phi;c_1,\ldots,c_{i-1})$, where $c_i = k$

$$x_i = G(z_i;\phi_k^g) \text{ so that } D(x_i;\phi_k^d) = 1 \quad (5)$$

where c_i now is an indicator variable, $\phi_{\{c_i=k\}} = [\phi_k^g, \phi_k^d]$ are the parameters of the kth GAN. Combining Equation 4-5 with 1-2, the inference of our new model becomes:

$$p(\phi|\Phi) = p(c_i|\mathbf{c}_{-\mathbf{i}}) \sim CRP(\alpha,\Phi,\mathbf{c}_{-\mathbf{i}}) \ c \in [1,K] \quad (6)$$

$$p(\phi_k^g|Z,\phi_k^d) \propto (\prod_{i=1}^{N_k^g} D(G(z_{(i)};\phi_k^g);\phi_k^d))p(\phi_k^g|\Phi^g) \quad (7)$$

$$p(\phi_k^d|Z,X,\phi_k^g) \propto \prod_{i=1}^{N_k^d} D(x_{(i)};\phi_k^d) \times \prod_{i=1}^{N_k^g}(1-$$

$$D(G(z_{(i)};\phi_k^g);\phi_k^d)) \times p(\phi_k^d|\Phi_d), \ 1 \le k \le K \quad (8)$$

where $\mathbf{c}_{-\mathbf{i}} = \{c_1,\ldots,c_{i-1}\}$. Sampling c in Equation 6 will naturally compute the right value for K, essentially conducting unsupervised clustering [44].

Classical GANs as maximum likelihood. Equation 6-8 is a Bayesian generalization of classic GANs. If a uniform prior is used for Φ and iterative maximum a posteriori (MAP) optimization is employed instead of sampling the posterior, then the local minima give the standard GANs [13]. However, even with a flat prior, there is a big difference between Bayesian marginalization over the whole posterior and approximating it with a point mass in MAP [52]. Equation 6-8 is a specification of Equation 1-2 with a CRP prior. Although Bayesian generalization over GANs have been explored before, we believe that this is first time a family of non-parametric Bayesian priors have been employed in modeling the distribution of GANs. Further, MIC-GANs aim to capture one cluster of images with one GAN in an unsupervised manner. The CRP prior can automatically infer the right value for K instead of pre-defining one as in existing approaches [33, 42, 15], where overestimating K will divide a cluster arbitrarily into several ones while underestimating K will mix multiple clusters.

3.3. Mixture of Infinite Conditional GANs

Iterative sampling over Equation 6-8 would theoretically infer the right values for ϕs and K. While ϕ^d and ϕ^g can be sampled [52] or approximated [13], K needs to be sampled indirectly by sampling c, which turns out to be extremely challenging. To see the challenges, we need to analyze the full conditional distribution of c. To derive the full distribution, we first give the conditional probability of c based on Eq. 4 as in [44]:

$$p(c_i = k|\mathbf{c_{-i}}) \propto \frac{N_k}{i - 1 + \alpha}$$
$$p(c_i \neq c_j \text{ for all } j < i|\mathbf{c_{-i}}) \propto \frac{\alpha}{i - 1 + \alpha} \quad (9)$$

Given the likelihood $p(x_i|z_i, \phi_k)$, or $p(x_i|\phi_k)$ if we omit z_i as it is independently sampled, we now combine it with Equation 9 to obtain the full conditional distribution of c_i given all other indicators and the current x_i:

$$p(c_i = k|\mathbf{c_{-i}}, x_i, \phi) \propto \frac{N_k}{i - 1 + \alpha} p(x_i|\phi_k) \text{ if } \phi_k \text{ exists,}$$
$$p(c_i = c_{new}|\mathbf{c_{-i}}, x_i, \phi) \propto \frac{\alpha}{i - 1 + \alpha} \int p(x_i|\phi)d\Phi$$
$$\text{if a new } \phi_{new} \text{ is needed} \quad (10)$$

where if a ϕ_{new} is needed then it will be sampled from the posterior $p(\phi|x_i, \Phi)$. Eq. 10 is used to sample Eq. 6.

3.3.1 Challenges of Inference

One method to sample c_i is Gibbs sampling [44], which requires the prior $p(\phi|\Phi)$ to be a conjugate prior for the likelihood; otherwise additional sampling (e.g. Metropolis-Hasting) is needed to approximate the integral in Equation 10. However, for MIC-GANs, not only is the prior not a conjugate prior for the likelihood, neither the likelihood nor the posterior can be even explicitly represented, which bring the following challenges: (1) Unable to directly compute the likelihood $p(x_i|\phi_c)$, which is a well-known issue for GANs [9]. (2) The sampling from $p(\phi|x_i, \Phi)$ is ill-conditioned. Since the prior Φ cannot be explicitly represented, direct sampling from $p(\phi|x_i, \Phi)$ becomes impossible. Alternatively, methods such as Markov Chain Monte Carlo are theoretically possible. But the dimension is often high, which will make the sampling prohibitively slow. (3) Sampling will dynamically change K. Each time K grows/shrinks, a GAN needs to be created/destroyed, which is far from ideal in terms of the training speed. This is also an issue of existing methods with multiple GANs [30, 17].

3.3.2 Enhanced Model for Inference

To tackle challenge (2), we introduce a conditional variable C while forcing all GANs to share the same ϕ so that they become $G_{\phi^g}(X|Z, C_k)$ and $D_{\phi^d}([0, 1]|X, C_k)$ instead of $G(X|Z; \phi_k^g)$ and $D([0, 1]|X, \phi_k^d)$ respectively, where $C \sim p(C)$ is well-behaved, e.g. a multivariate Gaussian. This formulation is similar to Conditional GANs (CGANs) but with a Bayesian treatment on C. Indeed, by introducing a conditional variable into multiple layers in the network, we exploit its ability of changing the mapping. Also, we now only need one GAN parameterized by $\phi = [\phi^g, \phi^d]$ and eliminate the need for multiple GANs without compromising the ability of learning K distinctive mappings. Now the role of C is the same as Φ in Equation 5, leading to:

$$\text{sample } z_i \sim Z, \{C_0, \ldots, C_k, \ldots, C_K\} \sim p(C)$$
$$\text{sample } c_i \sim CRP(\alpha, C; c_1, \ldots, c_{i-1}), \text{ where } c_i = k$$
$$x_i = G_{\phi^g}(z_i, C_k) \text{ so that } D_{\phi^d}(x_i, C_k) = 1, \quad (11)$$

where sampling from the posterior $p(\phi|x_i, C)$ (previously $p(\phi|x_i, \Phi)$) in Equation 10 becomes feasible. Note our formulation is different from traditional CGANs and GANs with multiple generators [17, 53, 28, 33] in: (1) our approach is still Bayesian. (2) we still model an infinite number of GANs and do not rely or impose assumptions on the prior knowledge of cluster numbers. (3) we do not need to actually instantiate multiple GANs.

Next, we still need to be able to compute the likelihood $p(x_i|\phi_k)$ in Equation 10, which is challenge (1). Now $p(x_i|\phi_k)$ becomes $p(x_i|C_k)$. Since GANs cannot directly compute likelihoods, we employ a surrogate model that can compute likelihoods while mimicking GAN's mapping behavior. Each C corresponds to one cluster, so the GANs can be seen as a mapping between Cs and images. This correspondence is exactly the same as classifiers. So we design a classifier Q to learn the mapping so that $p(x_i|C_k) \propto Q(c = k|x_i)$, where c is the same c as in Equation 10 but here is also a cluster label out of K clusters. Existing image classifiers can approximate likelihoods, e.g. through softmax. However, our experiments show that softmax-based likelihood approximation tends to behave like discriminative classifiers which focus on learning the classification boundaries. In Equation 10, we need a classifier with density estimation. We thus define Q as:

$$Q(c_i = k|x_i, \phi^q) \propto p(x_i|C_k, \phi^q)p(C_k|C)p(\phi^q|\Phi)$$
$$= \mathcal{N}(x_i|\mu_k, \Sigma_k, \phi^q)p(C_k|C)p(\phi^q|\Phi)$$
$$= \mathcal{N}(y_i|\mu_k, \Sigma_k)p(y_i|x_i, \phi^q)p(C_k|C)p(\phi^q|\Phi) \quad (12)$$

where $\mathcal{N}$ is a Normal distribution. μ_k and Σ_k are the mean and covariance matrix of the kth Normal. Q is realized by a deep neural net, parameterized by ϕ^q with a prior $p(\phi^q|\Phi)$, and classifies the latent code y_i of x_i by a Infinite Gaussian Mixture Model (IGMM) [50]. To see this is an IGMM, $p(C_k|C)$ is the same CRP as in Equation 11, so that now the

conditional variable C and the Gaussian component $[\mu, \Sigma]$ are coupled (through indices c under the same CRP prior). The inference on the Bayesian classifier Q can be done through iteratively sampling:

$$p(\phi^q|X, \mu, \Sigma) \propto \prod_{i=1}^{N} (\sum_{k=1}^{K} \mathcal{N}(y_i|\mu_k, \Sigma_k) p(y_i|x_i, \phi^q))$$
$$\times p(\phi^q|\Phi^q) \quad (13)$$
$$p(\mu_{c_i}, \Sigma_{c_i}|x_i, c_i = k, \mathbf{x_{-i}}, \mathbf{c_{-i}}, \phi^q) = \int\int p(y_i|\mu_k, \Sigma_k)$$
$$p(y_i|x_i, \phi^q) p(\mu_k, \Sigma_k|\mathbf{x_{-i}}, \mathbf{c_{-i}}) \partial\mu_k \partial\Sigma_k \quad (14)$$

where N is total number of images, K is total number of clusters, $\mathbf{x_{-i}}$ is all images except x_i with $\mathbf{c_{-i}}$ as their corresponding indicators. A Normal-Inverse-Warshart prior can be imposed on $[\mu, \Sigma]$ and we refer the readers to [50] for details. Alternatively, we could use IGMM directly on the images, but better classification results are achieved by classifying their latent features, as in the standard setting of many deep neural network based classifiers. We realize $p(y_i|x_i, \phi^q)$ as an encoder in Q.

Lastly for challenge (3), to avoid dynamically growing and shrinking K, we employ a truncated formulation inspired by [18], where an appropriate truncation level can be estimated. Essentially, the formulation requires us to start with a sufficiently large K then learn how many modes are actually needed, which is automatically computed due to the aggregation property of DPs. Note that the truncation is only for the inference purpose and does not affect the capacity of MIC-GAN to model an infinite number of GANs. We refer the readers to [18] for the mathematical proofs.

3.4. Adversarial Chinese Restaurant Process

Finally, we have our full MIC-GANs model (Eq. 11-12) and are ready to derive our new sampling strategy called Adversarial Chinese Restaurant Process (ACRP). A Bayesian inference can be done through Gibbs sampling: iteratively sampling on ϕ^g, ϕ^d, c (Equation 6-8,10), μ, Σ and ϕ^q (Equation 12-14). However, this tends to be slow given the high dimensionality. We thus propose to combine two schemes: optimization based on stochastic gradient descent and Gibbs sampling. While the former is suitable for finding the local minima that are equivalent to using flat priors and MAP optimization on $[\phi^g, \phi^d, \phi^q]$ [52], the latter is suitable for finding the posterior distributions of c, μ and Σ. We give the general sampling process of ACRP in Alg.1 and refer the readers to the supplementary materials for details.

In non-parametric Bayesian, $p(C)$ is governed by hyper-parameters which can be incorporated into the sampling. However, unlike traditional non-parametric Bayesian where a C would specify a generation process, our generation process is mainly learned through the GAN training. In other

words, the shape of $p(C)$ is not essential, as long as Cs are distinctive, i.e. conditioning different mappings. So we fix $p(C)$ to be a multivariate Gaussian without compromising the modeling power of MIC-GANs.

Finally, since the learned GANs are governed by a DP, another equivalent representation is $G(X|Z, C) = \sum_{k=1}^{\infty} \beta_k G_k(X|Z, C_k)$, where the subscript k indicates the kth GAN. β_k is a weight and $\sum_{k=1}^{\infty} \beta_k = 1$. This is another interpretation of DP called *stick-breaking* where $\beta_k = v_k \prod_{j=1}^{k-1} v_j$ and $v \sim Beta$ [55]. After learning, the weights βs can be computed by the percentage of images assigned to each GAN. The stick-breaking interpretation indicates that the learned Gs form a *convex combination* of clusters, which is an active construction of the latent space. This enables controlled generation e.g. using single Cs to generate images in different clusters, and easy exploration of the data space e.g. via interpolating Cs.

Algorithm 1 Adversarial Chinese Restaurant Process

Require:
    $epochs$ - the number of total training epochs;
    N is the total number of images;
    Initialize all variables (supplementary material);

1: **for** $epoch = 1$ to $epochs$ **do**
2:    For x_i in $\mathcal{X}$, classify x_i via Eq. 12;
3:    Compute $\{N_k\}_{k=1}^{K}$ in Eq. 10;
4:    For $k = 1$ to K Sample μ_k and Σ_k via Eq. 14;
5:    Sample $\{c_i\}_{i=1}^{N}$ via Eq. 10;
6:    Optimize ϕ^g and ϕ^d via conditional GAN loss;
7:    Optimize ϕ^q via Maximum Likelihood (Eq. 13);
8: **end for**

Implementation details. Due to the space limit, please refer to the supplementary materials for the details of models, data processing, training settings and performances.

4. Experiments

We adopt StyleGAN2 [24], StyleGAN2-Ada [22] and DCGAN [49] to validate that MIC-GANs can incorporate different GANs. We employ several datasets for extensive evaluation, including MNIST [29], FashionMNIST [58], CatDog from [6] and CIFAR-10 [27]. Moreover, we build a challenging dataset, named Hybrid, consisting of data with distinctive distributions. It is a mixture of the '0', '2', '4', '6', '8' from MNIST, the 'T-shirt', 'pullover', 'trouser', 'bag', 'sneaker' from FashionMNIST, the cat images from Catdog and human face images from CelebA [34].

For comparison, we employ as baselines several state-of-the-art GANs including DMGAN [25], InfoGAN [5], ClusterGAN [42], DeliGAN [15], Self-Conditioned GAN [33] and StyleGAN2-Ada [22], whose code is shared by the authors. Notably, DeliGAN, InfoGAN, ClusterGAN and Self-

	MNIST	FashionMNIST	CatDog	Hybrid
Purity	0.9489	0.6362	0.9736	0.9567
FID	12.79	17.97	26.24	11.2

Table 1. Purity and FID scores from MIC-GANs.

		MNIST			Hybrid	
	K	Purity	FID	K	Purity	FID
1	20	**0.9489**	12.79	25	**0.9567**	**11.2**
2	20	0.7612	<u>10.82</u>	25	0.8959	32.31
3	10	0.8589	11.9	12	0.881	71.84
4	10	<u>0.8784</u>	**9.27**	12	<u>0.9457</u>	<u>19.15</u>
5	10	n/a	127.58	10	n/a	241.69

Table 2. Purity and FID of MIC-GANs(1), DMGAN(2), Info-GAN(3), ClusterGAN(4) and DeliGAN(5).

Conditioned GAN need a pre-defined cluster number, while MIC-GANs and DMGAN compute it automatically. To be harsh on MIC-GANs, we provide a large cluster number as an initial guess to MIC-GANs and DMGAN, while giving the ground-truth class number to the other methods.

4.1. Unsupervised Clustering

Although MIC-GANs focus on unsupervised image generation, it clusters data during learning. We evaluate its clustering ability by Cluster Purity [35] which is a common metric to measure the extent to which clusters contain a single class: $Purity = \frac{1}{N}\sum_{i=1}^{K} max_j |c_i \cap t_j|$, where N is the number of images, K is the number of clusters, c_i is a cluster and t_j is the class which has the highest number of images clustered into c_i. The purity reaches the maximum value 1 when every image is clustered into its own class. Since MIC-GANs compute K clusters, we rank them by their β_ks in the descending order and choose the top n clusters for purity calculation, where n is the ground-truth class number. Note the ground-truth is only used in the testing phase, not in the training phase.

We use K=20, 20, 4 and 25 for MINST, FashionMNIST, CatDog and Hybrid. The results are shown in Table 1. MIC-GANs achieve high purity on MNIST, CatDog and Hybrid. On FashionMNIST, the purity is relatively low. We find that it is mainly due to the ambiguity in the class labels. One example is that the bags with/without handles are given the same class label in FashionMNIST but divided by MIC-GANs into two clusters, as shown later. This finer-level clustering is automatically conducted, which we argue is still reasonable albeit leading to a lower purity score.

As a comparison, we also evaluate the baseline methods on MNIST and Hybrid, with the ground-truth class number K=10 for InfoGAN and ClusterGAN, and K=20 for DMGAN. Note that DeliGAN does not have a clustering module, so we exclude it from this evaluation. For both MIC-GANs and DMGAN, we use the top n clusters where n is the ground-truth class number. As we can see

in Table 2, surprisingly, MIC-GANs achieve the highest purity score (0.9489) among these methods, even without using the ground-truth class number in training. It demonstrates that the MIC-GANs can effectively learn the modes in the MNIST without any supervision. In addition, we also conduct the comparison on Hybrid, with the ground-truth K=12 for InfoGAN and ClusterGAN, and K=25 for DMGAN. As shown in Table 2, MIC-GANs achieve the best purity score (0.9567). ClusterGAN (0.9457) comes a close second. However, we provide the ground-truth class number to ClusterGAN which is strong prior knowledge, while MIC-GANs have no such information.

4.2. Image Generation Quality

We conduct both qualitative and quantitative evaluations on the generation quality. For quantitative evaluations, we use Frechet Inception Distance (FID) [16] as the metric. Qualitatively, generated samples can be found in Figure 1-2. Due to the space limit, we rank the modes based on their βs in the descending order and only show the top modes. More results can be found in the supplementary materials. Intuitively, all the modes are captured cleanly, shown by that the top modes contain all the classes in the datasets. This is where MIC-GANs capture most of the 'mass' in the datasets. In addition, each mode fully captures the variation within the mode, no matter it is the writing style in MNIST, the shape in FashionMNIST or the facial features in CatDog and CelebA. The balance between clustering and within-cluster variation is automatically achieved by MIC-GANs in an unsupervised setting. This is very challenging because the within-cluster variations are distinctive in Hybrid since the data comes from four datasets. Beyond the top modes, the low-rank modes also capture information. But the information is less meaningful (the later modes in MNIST) or mixed (later modes in Hybrid) or contain subcategories such as the separation of bags with/without handles in FashionMNIST. However, this does not undermine MIC-GANs because the absolute majority of the data is captured in the top modes. Quantitatively, we found that good FID scores can be achieved, shown in Table 1.

We further conduct comparisons on MNIST and Hybrid using the same settings as above for all methods. As demonstrated in Table 2, MIC-GANs obtain a comparable FID score to other methods without any supervision in MNIST. Besides DeliGAN, DMGAN and MIC-GANs achieve slightly worse FID scores. We speculate that this is because MIC-GANs and DMGANs do not use prior knowledge and therefore have a disadvantage. One exception is DeliGAN. In their paper, the authors chose a small dataset (500 images) for training and achieved good results. However, when we run their code on the full MNIST dataset, we were not able to reproduce comparable results even after trying our best to tune various parameters. Next, in the chal-
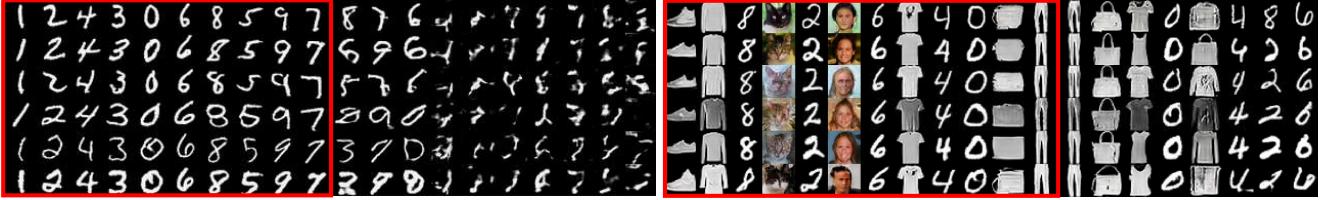
Figure 1. Our results on MNIST (left) and Hybrid (right) dataset, both with $K = 25$. Each column is generated from a mode, and the columns are sorted by αs (the last 4 modes are not shown). The red boxes mark the top n modes in the results.
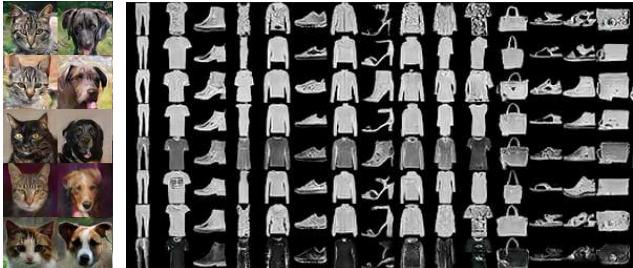
Figure 2. Our results on CatDog (left) and FashionMNIST (right) dataset. Each column is generated from one mode, and the columns are sorted by αs.

Ours			SC-GAN			StyleGAN2-Ada		
C	K	FID	C	K	FID	C	K	FID
4	10	**5.31**	4	10	237.96	4	-	5.64
7	15	**5.09**	7	15	41.30	7	-	5.16
10	20	**4.72**	10	20	22.00	10	-	5.03

Table 3. Comparisons of our method, SC-GAN (Self-Conditioned GAN), and Styelgan2-Ada on CIFAR. 'C' means the ground-truth class number in the dataset.

lenging Hybrid dataset whose distribution is more complex than MNIST, MIC-GANs achieve the best FID score. Without any supervision, MIC-GANs not only capture the multiple data modes well, but generate high-quality images. We also conduct comparisons on CIFAR with Self-Conditioned GAN [33] and StyleGAN2-Ada [22]. To investigate how MIC-GANs compare with other methods on a dataset with different numbers of modes, we sample C={4,7,10} classes from CIFAR where C=10 is the full dataset. We also adopt StyleGAN2-Ada in MIC-GANs for CIFAR. The results show that MIC-GANs can achieve better FID scores (Table 3). The change of baselines is mainly due to that we only compare MIC-GANs with methods on the datasets on which they are tested, for fairness.

As a visual comparison, we show the generated images from different GANs in Figure 3. In the top row, MIC-GANs generate perceptually comparable results on MNIST to InfoGAN and ClusterGAN which were fed with the ground-truth class number, while achieving better clustering than DMGAN (e.g., mixed '9' and '4', '9' and '7') which is also unsupervised. In the challenging Hybrid dataset (bottom), MIC-GANs are able to generate high-quality images while correctly capturing all the modes. CIFAR images are shown in the supplementary materials.

		DCGAN			StyleGAN2	
	K	Purity	FID	K	Purity	FID
MNIST	1	-	9.89	1	-	12.96
	15	0.9384	5.22	15	0.9397	9.92
	20	0.9578	8.62	20	0.9489	12.79
	25	0.93	9.13	25	0.9487	11.9
	K	Purity	FID	K	Purity	FID
Hybrid	1	-	60.83	1	-	15.47
	15	0.9218	50.74	15	0.942	15.7
	20	0.9611	48.17	20	0.923	13.31
	25	0.966	45.07	25	0.9567	11.2

Table 4. Purity and FID of ablation studies on MNIST and Hybrid.

4.3. Ablation Study

We conduct ablation studies to test MIC-GANs' sensitivity to the free parameters. As an unsupervised and non-parametric method, there are not many tunable parameters which is another advantage of MIC-GANs. The main parameters are the K value and the GAN architecture. We therefore test another popular GAN, DCGAN [49] and vary the K value. As shown in Table 4, the purity scores are very similar, which means the clustering is not significantly affected by the choices of GANs or the K value. FID scores vary across datasets, which is mainly related to the specific choice of the GAN architecture. However, stable performance is obtained across different Ks in every setting. In addition, the FID scores when $K = 1$ are in general worse than those when $K > 1$, confirming that our method can mitigate mode collapses. The same mode collapse mitigation can also be observed when comparing our method with StyleGAN2-Ada on CIFAR-10 (Table 3), where StyleGAN2-Ada is just our method with $K = 1$.

4.4. Benefits of Non-parametric Learning

In real-world scenarios, we do not often know the cluster number *a priori*, under which we investigate the performance of InfoGAN, ClusterGAN and DeliGAN. We use Hybrid and run experiments with K=8, 12, 16 and 22 to cover K values that are smaller, equal to and bigger than the ground-truth K=12. We only show the results of K=8 in Figure 4 and refer the readers to the supplementary materials for fuller results and analysis. Intuitively, when K is smaller than the ground-truth, the baseline methods either cannot capture all the modes or capture mixed modes; when

Figure 3. Generation comparisons on MNIST (top) and Hybrid (bottom) dataset. We use the ground-truth $K = 10$ on MNIST and $K = 12$ on Hybrid for InfoGAN, ClusterGAN and DeliGAN, and $K = 20$ for MNIST and $K = 25$ for Hybrid for MIC-GANs and DMGAN. Each column is generated from a mode for MNIST (top), and each row is generated from a mode for Hybrid (bottom).
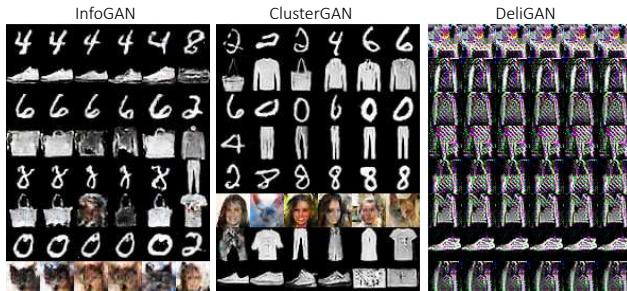

Figure 4. $K = 8$ results of InfoGAN, ClusterGAN and DeliGAN.

K is larger than the ground-truth, they capture either mixed modes or repetitive modes. In contrast, although MIC-GANs (Figure 1-2) also learn extra modes, it concentrates the mass into the top modes resulting in clean and complete capture of modes. MIC-GANs are capable of high-quality image generation and accurate data mode capturing, while being robust to the initial guess of K.

4.5. Latent Structure

Since MIC-GANs are a convex combination of GANs, we can do controlled generation, including using a specific mode, and interpolating between two or more different modes for image generation. Figure 1-2 already show image generation based on single modes. We show interpolations between two Cs and among four Cs respectively in Figure 5. Through both bi-mode and multi-mode interpolation, we show that MIC-GANs structure the latent space well so that smooth interpolations can be conducted within the subspace bounded by the base modes.
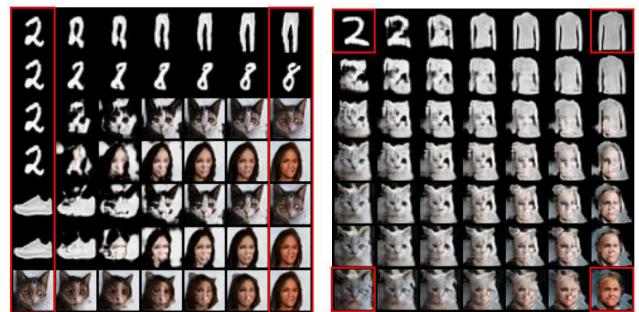
Figure 5. Left: each row is the interpolation results between two latent codes, where the first column and the last column are the original images. Right: the interpolation results among four latent codes, where each corner represents one mode.

5. Conclusion

We proposed a new unsupervised and non-parametric generative framework MIC-GANs, to jointly tackle two fundamental GAN issues, mode collapse and unstructured latent space, based on parsimonious assumptions. Extensive evaluations and comparisons show that MIC-GANs outperform state-of-the-art methods on multiple datasets. MIC-GANs do not require strong prior knowledge, nor do they need much human intervention, providing a robust and adaptive solution for multi-modal image generation.

References

[1] David J. Aldous. Exchangeability and related topics. In P. L. Hennequin, editor, *École d'Été de Probabilités de Saint-Flour XIII — 1983*, pages 1–198, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

[2] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML 2017*, 2017.

[3] Matan Ben-Yosef and Daphna Weinshall. Gaussian mixture generative adversarial networks for diverse datasets, and the unsupervised clustering of images. *CoRR*, abs/1808.10356, 2018.

[4] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. In *ICLR 2017*, 2017.

[5] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS 2016*, pages 2172–2180, 2016.

[6] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR 2020*, pages 8188–8197, 2020.

[7] Ishan P. Durugkar, Ian Gemp, and Sridhar Mahadevan. Generative multi-adversarial networks. In *ICLR 2017*, 2017.

[8] Amine Echraibi, Joachim Flocon-Cholet, Stéphane Gosselin, and Sandrine Vaton. On the variational posterior of dirichlet process deep latent gaussian mixture models. *CoRR*, abs/2006.08993, 2020.

[9] Hamid Eghbal-zadeh and G. Widmer. Likelihood estimation for generative adversarial networks. *ArXiv*, abs/1707.07530, 2017.

[10] Hamid Eghbal-zadeh, Werner Zellinger, and Gerhard Widmer. Mixture density generative adversarial networks. In *CVPR 2019*, pages 5820–5829, 2019.

[11] Thomas S. Ferguson. A bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[12] Arnab Ghosh, Viveka Kulharia, Vinay P. Namboodiri, Philip H. S. Torr, and Puneet Kumar Dokania. Multi-agent diverse generative adversarial networks. In *CVPR 2018*, pages 8513–8521, 2018.

[13] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks, 2017.

[14] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In *NIPS 2017*, pages 5767–5777, 2017.

[15] Swaminathan Gurumurthy, Ravi Kiran Sarvadevabhatla, and R. Venkatesh Babu. Deligan: Generative adversarial networks for diverse and limited data. In *CVPR 2017*, pages 4941–4949, 2017.

[16] 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 *NIPS 2017*, volume 30, pages 6626–6637, 2017.

[17] Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. MGAN: Training generative adversarial nets with multiple generators. In *ICLR 2018*, 2018.

[18] Hemant Ishwaran and Lancelot F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, 2001.

[19] Abdul Jabbar, Xi Li, and Bourahla Omar. A survey on generative adversarial networks: Variants, applications, and training, 2020.

[20] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. In *IJCAI 2017*, pages 1965–1972, 2017.

[21] Weonyoung Joo, Wonsung Lee, Sungrae Park, and Il-Chul Moon. Dirichlet variational autoencoder. *Pattern Recognition*, 107:107514, 2020.

[22] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *NIPS 2020*, 2020.

[23] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR 2019*, pages 4401–4410, 2019.

[24] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR 2020*, pages 8107–8116, 2020.

[25] Mahyar Khayatkhoei, Maneesh K Singh, and Ahmed Elgammal. Disconnected manifold learning for generative adversarial networks. In *NIPS 2018*, pages 7343–7353, 2018.

[26] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR 2014*, 2014.

[27] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[28] J. N. Kundu, M. Gor, D. Agrawal, and V. B. Radhakrishnan. Gan-tree: An incrementally learned hierarchical generative framework for multi-modal data distributions. In *ICCV 2019*, pages 8190–8199, 2019.

[29] Yann LeCun, Corinna Cortes, and Christopher J. C. Burges. The mnist database of handwritten digits, 1998. `http://yann.lecun.com/exdb/mnist/`.

[30] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning, 2020.

[31] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabas Poczos. Mmd gan: Towards deeper understanding of moment matching network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NIPS 2017*, volume 30. Curran Associates, Inc., 2017.

[32] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *NIPS 2016*, pages 469–477, 2016.

[33] Steven Liu, Tongzhou Wang, David Bau, Jun-Yan Zhu, and Antonio Torralba. Diverse image generation via self-conditioned gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14286–14295, 2020.

[34] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV 2015*.

[35] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, USA, 2008.

[36] Qi Mao, Hsin-Ying Lee, Hung-Yu Tseng, Siwei Ma, and Ming-Hsuan Yang. Mode seeking generative adversarial networks for diverse image synthesis. In *CVPR 2019*, pages 1429–1437, 2019.

[37] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV 2017*, pages 2813–2821, 2017.

[38] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. In *ICLR 2017*, 2017.

[39] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[40] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR 2018*, 2018.

[41] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *ICLR 2018*, 2018.

[42] Sudipto Mukherjee, Himanshu Asnani, Eugene Lin, and Sreeram Kannan. Clustergan: Latent space clustering in generative adversarial networks. In *AAAI 2019*, volume 33, pages 4610–4617, 2019.

[43] Eric T. Nalisnick and Padhraic Smyth. Stick-breaking variational autoencoders. In *ICLR 2017*, 2017.

[44] Radford M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

[45] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *NIPS 2016*, pages 271–279. 2016.

[46] Augustus Odena. Open questions about generative adversarial networks. In *distill.pub, 2019*, 2019.

[47] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML 2017*, ICML'17, page 2642–2651, 2017.

[48] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI 2018*, 2018.

[49] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR 2016*, 2016.

[50] Carl Edward Rasmussen. The infinite gaussian mixture model. In *NIPS 1999*, NIPS'99, page 554–560, Cambridge, MA, USA, 1999. MIT Press.

[51] Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, and Shakir Mohamed. Variational approaches for auto-encoding generative adversarial networks. *CoRR*, abs/1706.04987, 2017.

[52] Yunus Saatchi and Andrew Gordon Wilson. Bayesian gan. *NIPS 2017*, 2017-December:3623–3632, 2017. 31st Annual Conference on Neural Information Processing Systems, NIPS 2017 ; Conference date: 04-12-2017 Through 09-12-2017.

[53] A. Sage, R. Timofte, E. Agustsson, and L. V. Gool. Logo synthesis and manipulation with clustered generative adversarial networks. In *CVPR 2018*, pages 5879–5888, 2018.

[54] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In *NIPS 2016*, pages 2234–2242. 2016.

[55] Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica Sinica*, 4(2):639–650, 1994.

[56] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U. Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *NIPS 2017*, page 3310–3320, 2017.

[57] Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pages 1214–1223. PMLR, 2018.

[58] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[59] Dingdong Yang, Seunghoon Hong, Yunseok Jang, Tianchen Zhao, and Honglak Lee. Diversity-sensitive conditional generative adversarial networks. In *ICLR 2019*, 2019.

[60] Junbo Zhao, Michaël Mathieu, and Yann LeCun. Energy-based generative adversarial networks. In *ICLR 2017*, 2017.

A. Algorithm Details

The training of MIC-GANs is split into two stages, the initialization stage and the Adversarial Chinese Restaurant Process (ACRP) Stage.

A.1. Initialization Stage

The initialization stage is to initialize the generator, enabling it to produce images of good quality. At the same time, we require the generator to produce conditioned output without supervised class labels.

The detailed algorithm for initialization is shown in Algorithm 2. The training procedure is the same as the ordinary GANs, except that the generator is given a conditioned input. Note that the discriminator is not conditional, so it is not a conditional GAN. $Categ(\alpha_1, ..., \alpha_K)$ is the category distribution where index number k is sampled according to the probability proportional to α_k. The conditional input for generator is uniformly sampled, i.e. $\alpha_1, ..., \alpha_K = \frac{1}{K}$.

After initialization, the generator can produce conditioned outputs. Generally, the outputs from one condition are more likely to be close to one class of images. However, because K is different from the ground-truth class number, the outputs from one condition often either contain only a part of one class or multiple classes. Taking MNIST for example, when using $K = 20$ which is larger than the ground-truth class number, after initialization there may be two modes generating '6' and one mode generating both '7' and '9'. However, it will be resolved in the later ACRP stage.

A.2. ACRP Stage

The main algorithm of ACRP is shown in Algorithm 3, and the Chinese Restaurant Process Sampling algorithm is shown in Algorithm 4. In practice, the encoder network in Q is initialized before training in every epoch to prevent overfitting. The parameters of the GMMs need to be initialized before training. We let the covariance matrices Σ_ks be the identity matrix, and initialize the mean μ_ks in the way that they become the vertices of a high-dimensional simplex and are equidistant to each other. This is to ensure that the Gaussians are distinctive. Next, the dimensionality of the latent space of the encoder in Q needs to be decided. Theoretically, it can be any dimension that is smaller than that of the data space. In practice, we set the dimension of both the image embedding and GMM to K which is the number of modes, so that conveniently the μ_ks are the basis vectors of such a K-dimensional space. One straightforward solution is to use the one-hot K-vectors as the μ_ks' initialization. As for the encoder loss $\mathcal{L}_Q(e, \mu_c)$, we maximize the log likelihood of e with respect to the Gaussian with μ_c as its mean.

Likelihoods in GANs. We do not solve the likelihood problem of GANs directly. MIC-GANs employ a surrogate

Variable	Meaning
$\mathcal{X}$	the whole real images
K	the number of modes
N	the total number of real images
G_{ϕ^G}	generator parameterized by ϕ^G
D_{ϕ^D}	discriminator parameterized by ϕ^D
Q_{ϕ^Q}	classifier parameterized by ϕ^Q
z	the input noise for generator
α_k	the sampling probability of mode k
c_i	the picked mode index for each real image i
μ_k, Σ_k	the parameters of the kth Gaussian
N_k	the number of real images associated with mode k
$p_{i,k}$	the likelihood of real image i on mode k
e	the embedding of an image, from encoder Q

Table 5. Symbols of MIC-GANs Training

density estimator. The design is due to the following reasons. First, MIC-GANs are designed to work essentially with any GANs. So the density estimation needs to be independent of the specific GAN architecture. Also, different GANs are designed for different tasks, e.g. StyleGAN, Big-GAN, etc. They all contain specific architectures optimized for their aimed tasks. Therefore, we choose to keep any chosen GAN intact under MIC-GANs. Employing a surrogate density estimator is our current solution, and we are actively looking for a 'true' solution.

To help understand ACRP, we present a visualization (Figure 6) of the procedure of our algorithm with a simple dataset which only consists of number '7' and number '9' from MNIST. The figure visualizes line 7-18 of Algorithm 3. In the algorithm, we set the embedding dimension to be 2 with $K = 3, iters_1 = 1, iters_2 = 1$, and we fix the means of GMMs to be three vertices of an equilateral triangle for better visualization quality. As shown in Figure 6, at first, 'mode 1' contains most '9's and 'mode 2' contains most '7's while 'mode 3' contains both '9's and '7's. With the algorithm progressing, the number of images classified to 'mode 3' is gradually reduced, because more '9's that were originally classified to 'mode 3' are now classified to 'mode 1', and similarly '7's that are originally under 'mode 3' are now classified to 'mode 2'. In Epoch 4, most of the images are divided into two modes and the classification is almost correct. Meanwhile, the original 'mode 3' basically disappears as the probability of it being sampled again becomes nearly zero. To see this, N_3 in line 12 in Algorithm 3 after CRP becomes very small and the probability of 'mode 3' being sampled again is proportional to N_3. Figure 6 shows the 'richer gets richer' property of Chinese Restaurant Process.

Algorithm 2 Initialization

Require:
  $epochs$ - the number of total training epochs;
  N_{init} - the number of images for initialization training;

1: **for** $n = 1$ to N_{init} **do**
2:      Sample $x \sim \mathcal{X}$
3:      Sample $z \sim \mathcal{N}(0, 1)$, $c \sim Categ(\alpha_1, ..., \alpha_K)$
4:      Generate fake image $\hat{x} = G(z, c)$
5:      Optimize ϕ^g and ϕ^d via GAN loss
6: **end for**

Algorithm 3 Adversarial Chinese Restaurant Process

Require:
  $epochs$ - the number of total training epochs;
  N_Q - the number of images for training encoder in each epoch;
  N_{GD} - the number of images for training generator and discriminator in each epoch;
  $iters_1$ - the number of iterations for CRP sampling and GMM updating;
  Initialize() ;

1: **for** $epoch = 1$ to $epochs$ **do**
2:      **for** $n = 1$ to N_Q **do**   ▷ Train Q
3:          Sample $z \sim \mathcal{N}(0, 1)$, $c \sim Categ(\alpha_1, ..., \alpha_K)$
4:          Get embedding $e = Q(G(z, c))$
5:          Optimize ϕ_Q via encoder loss $L_Q = \mathcal{L}_Q(e, \mu_c)$
6:      **end for**
7:      **for** $iter = 1$ to $iters_1$ **do**   ▷ Classify x
8:          **for** x_i in $\mathcal{X}$ **do**   ▷ Computing likelihood
9:              $e_i = Q(x_i)$
10:             $p_{i,k} = Gauss(e_i | \mu_k, \Sigma_k)$ for $k = 1$ to K
11:          **end for**
12:          Sample $\{c_i\}_{i=1}^N$, $\{N_k\}_{k=1}^K$ via CRP (Alg. 4)
13:          $\mathbb{E}_k \leftarrow \emptyset$ for $k = 1$ to K
14:          $\mathbb{E}_{c_i} \leftarrow \mathbb{E}_{c_i} \bigcup \{e_i\}$ for each e_i
15:          **for** $k = 1$ to K **do**   ▷ Update GMMs
16:             Update μ_k and Σ_k with $\mathbb{E}_k$
17:          **end for**
18:      **end for**
19:      $\alpha_k \leftarrow \dfrac{N_k}{N}$
20:      **for** $n = 1$ to N_{GD} **do**   ▷ Train GAN
21:          Sample $x_i \sim \mathcal{X}$, and fetch corresponding c_i
22:          Sample $z \sim \mathcal{N}(0, 1)$, $c \sim Categ(\alpha_1, ..., \alpha_K)$
23:          Generate fake image $\hat{x} = G(z, c)$
24:          Optimize ϕ^g and ϕ^d via conditional GAN loss
25:      **end for**
26: **end for**

▲ : 7   ● : 9
Blue: mode 1   Orange: mode 2   Green: mode 3
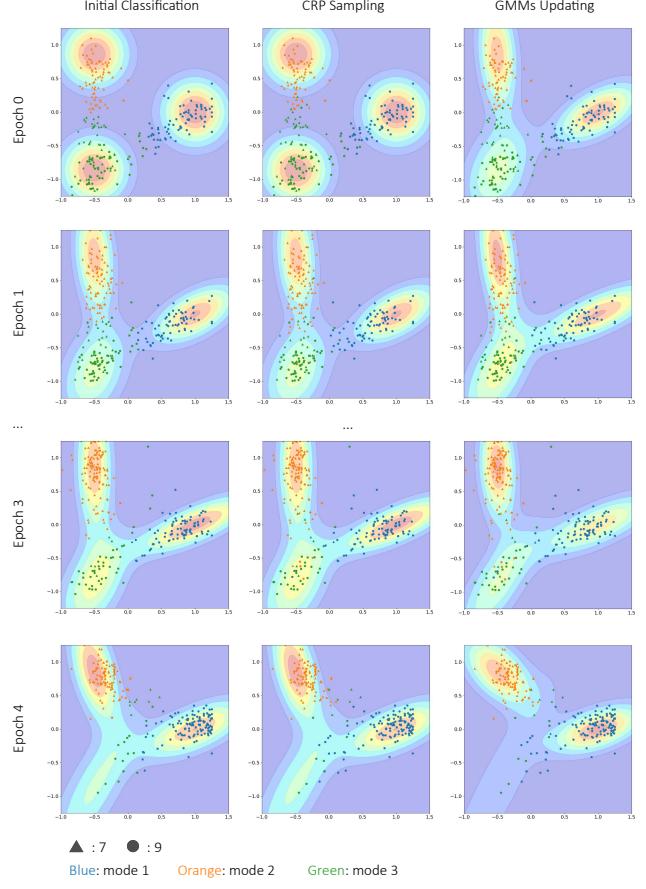
Figure 6. The transformation of classification results and GMMs in the ACRP stage. In each small figure, the small dots represent embeddings of training images from encoder Q, with the triangle dots representing number '7' and the circle dots representing number '9'. The colors of the dots represent the classifications to three modes. The background color visualizes the shape of GMMs. In each epoch, the left two figures show the classification results conducted directly from the gaussian probability and after CRP sampling and the right figure shows the updated GMMs based on the classification results after CRP sampling.

B. Implementation Details

B.1. Network Architecture

We adopt different GAN models including DC-GAN [49], StyleGAN2 [24] and StyleGAN2-Ada [22] to validate our algorithm. Specifically, in order to achieve conditioned generation, we modify the input of the generators to take conditions. The detailed implementation of the conditional inputs is shown in Figure 7. For DCGAN and StyleGAN2-Ada, the condition of the generator is specified by adding the conditioned latent code C_c to the noise z. In StyleGAN2, we tried to control the condition by picking one of K constant inputs for the synthesis network.

The network architecture of discriminator needs to be handled differently in different stages because the discrim-

Algorithm 4 Chinese Restaurant Process Sampling

Require:

$iters_2$ - the number of iterations for CRP sampling;

1: $N_k \leftarrow 0$ for $k = 1$ to K
2: **for** x_i in $\mathcal{X}$ **do**
3: $\quad c_i \leftarrow argmax(\{p_{i,k}\}_{k=1}^K)$
4: $\quad N_{c_i} = N_{c_i} + 1$
5: **end for**
6: **for** $iter = 1$ to $iters_2$ **do**
7: $\quad$ **for** x_i in $\mathcal{X}$ **do**
8: $\quad\quad N_{c_i} = N_{c_i} - 1$
9: $\quad\quad \beta_k \leftarrow N_k \cdot p_{i,k}$ for $k = 1$ to K
10: $\quad\quad \beta_k \leftarrow \dfrac{\beta_k}{\sum \beta_k}$ for $k = 1$ to K
11: $\quad\quad$ Sample $c_i \sim Categ(\beta_1, ..., \beta_K)$
12: $\quad\quad N_{c_i} = N_{c_i} + 1$
13: $\quad$ **end for**
14: **end for**

---

inator needs to take conditions in ACRP stage but the conditions are not reliable in the initialization stage. So we keep the discriminator as the original one in DCGAN or StyleGAN2 during initialization, and in ACRP stage, modify it to take conditions. For the discriminator of DC-GAN and StyleGAN2, we follow the approach of traditional cGAN [39], and for the discriminator of StyleGAN2-Ada, we follow the approach in [41]. After initialization and at the beginning of the ACRP stage, a condition input is added to the discriminator.

For the network architecture of encoder in Q, we simply adopt a multi-layer convolutional network. For the dataset of MNIST, FashionMNIST, and Hybrid, we use a 4-layer CNN with a fully connected output layer, and for the dataset of CatDog, CIFAR and Tiny Imagenet, we use a 7-layer CNN with a fully connected output layer. Both of them do not use BatchNorm Layer.

Besides the DCGAN and StyleGAN, there are other GANs that are also suitable for mode separation, e.g., FiLM [48].In fact, our algorithm can be applied to any conditional GANs theoretically.

B.2. Training Details

Images in MNIST, FashionMNIST, Hybrid, CIFAR and Tiny Imagenet are resized to 32, and images in CatDog are resized to 64. When training, the batch size is set to 64 for CatDog and CIFAR, and 256 for the other datasets. During initialization, N_{init} is set to $2400k$ for the MNIST FashionMNIST, Hybrid dataset, $1000k$ for the CatDog dataset, $2000k$ for the CIFAR and Tiny Imagenet dataset. In ACRP stage, N_Q is set to $64k$, and N_{GD} is set to $300k$ for the MNIST FashionMNIST, Hybrid dataset, $100k$ for the CatDog dataset, $200k$ for the CIFAR and Tiny Imagenet

(a) DCGAN   (b) StyleGAN2   (c) StyleGAN2-Ada
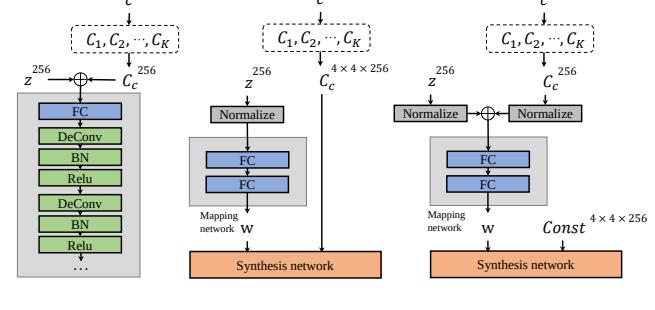
Figure 7. The conditional input heads of the generators for DC-GAN, StyleGAN2 and StyleGAN2-Ada.

	training Q	sampling	training GAN
DCGAN	0.5mins	2.6mins	4.5mins
StyleGAN2	0.5mins	2.6mins	15mins

Table 6. Training time distribution for one epoch on MNIST.

epochs	1	5	9	13	19
purity	0.839	0.908	0.911	0.927	0.929

Table 7. Purity vs sampling epochs on MNIST with $K = 15$.

dataset. We trained the MIC-GANs for totally 40 epochs in ACRP stage, as the classification results of the real images converge quickly, we stop CRP Sampling (re-classification) after 10 epochs, so the GAN can focus on improving the quality of image generation.

For the dataset of Tiny ImageNet, we picked 10 classes for the MIC-GANs training, which are 'goldfish', 'black widow', 'brain coral', 'golden retriever', 'monarch', 'beach wagon', 'beacon', 'bullet train', 'triumphal arch', 'lemon'.

C. Quantitative Results

Table. 6 shows the training time distribution for one epoch on MNIST dataset. We find that the sampling in learning the prior is not the most time-consuming component, while the training of the GANs itself dominates the training time. And the situation is similar on all datasets.

In Table. 7, we show the relationship between the purity and sampling epochs on the MNIST dataset. We find that the purity converges quickly in the first few epochs (similar on other datasets). So we stop the CRP sampling after 10 epochs and use the stable classification results for GAN training.

D. Generation Results

Figure 8 visualizes the MNIST and Hybrid results of our method and DMGAN [25]. We can find that even using different Ks, our method can provide stable results, which demonstrates the ability of unsupervised clustering of our method. DMGAN achieves a similar effect as ours, but it often learns mixed modes, e.g., the confusion between '4' and '9' on MNIST. Furthermore, our method is flexible with the architecture of GAN without compromising the train-

ing speed much, which means that we can employ complex GANs such as StyleGAN2 on complex datasets. However, it will be prohibitively expensive for DMGAN to achieve the same because DMGAN requires K generators for K modes, while MIC-GANs only require K latent codes.

Figure 9 shows the results of InfoGAN [5], Cluster-GAN [42] and DeliGAN [15] on Hybrid with different Ks. Obviously, these methods fail to perform correct clustering when the ground-truth $K = 12$ is unknown and the best way is to make multiple guesses. However, when $K < 12$, there will be mixed modes; when $K > 12$, there will be repetitive modes, as well as mixed modes. This shows that these methods either cannot produce good results or require a large number of guesses in the absence of the ground-truth, while MIC-GANs can generate satisfying results in one run.

Figure 10 shows the generation results of our method, Self-Conditioned GAN [33] and StyleGAN2-Ada [22] on CIFAR with different Cs and Ks. CIFAR is a difficult dataset for generation, and it is an even more challenging dataset for conditioned generation based on unsupervised clustering. In our method, some modes can generate images that are from clear-cutting single classes, e.g., 'automobile', 'airplane', 'horse'. In other cases, images generated from one mode consist of images from two or more classes. This reflects the fact that images can be clustered based on different criteria. This sometimes leads to different classification results between MIC-GANs and human labels. For example, images can be classified according to the colors or shapes or semantics. While human labels in CIFAR are primarily based on semantics (object identities), it is normal that MIC-GANs at times generate images from one mode that match several ground-truth classes. Nevertheless, we can still find some interesting similarities among the images generated from one mode. In addition, MIC-GANs improve the generation quality in general with lower FID scores shown in the paper. We also find that Self-Conditioned GAN suffers from mode collapse in several modes and the problem gets worse when K is small. StyleGAN2-Ada is able to generate images with diversities but ours still achieve better FID scores.

Figure 11 shows the generation results of our method on the Tiny Imagenet dataset. Without any class supervision, our algorithm still generates several reasonable conditional results. For example, line 1, 2, 3, 5, 11, 12 correctly produce the images of lemon, triumphal arch, beacon, brain coral, monarch and beach wagon, while several modes generate the mixtures of classes, like line 4 and 7. Another interesting observation is that line 9 mostly generates bullet trains facing the right while line 10 genearates buleet trains facing the left. Tiny Imagenet is a difficult dataset, so the generation is less ideal on some modes, but still covers most of them.
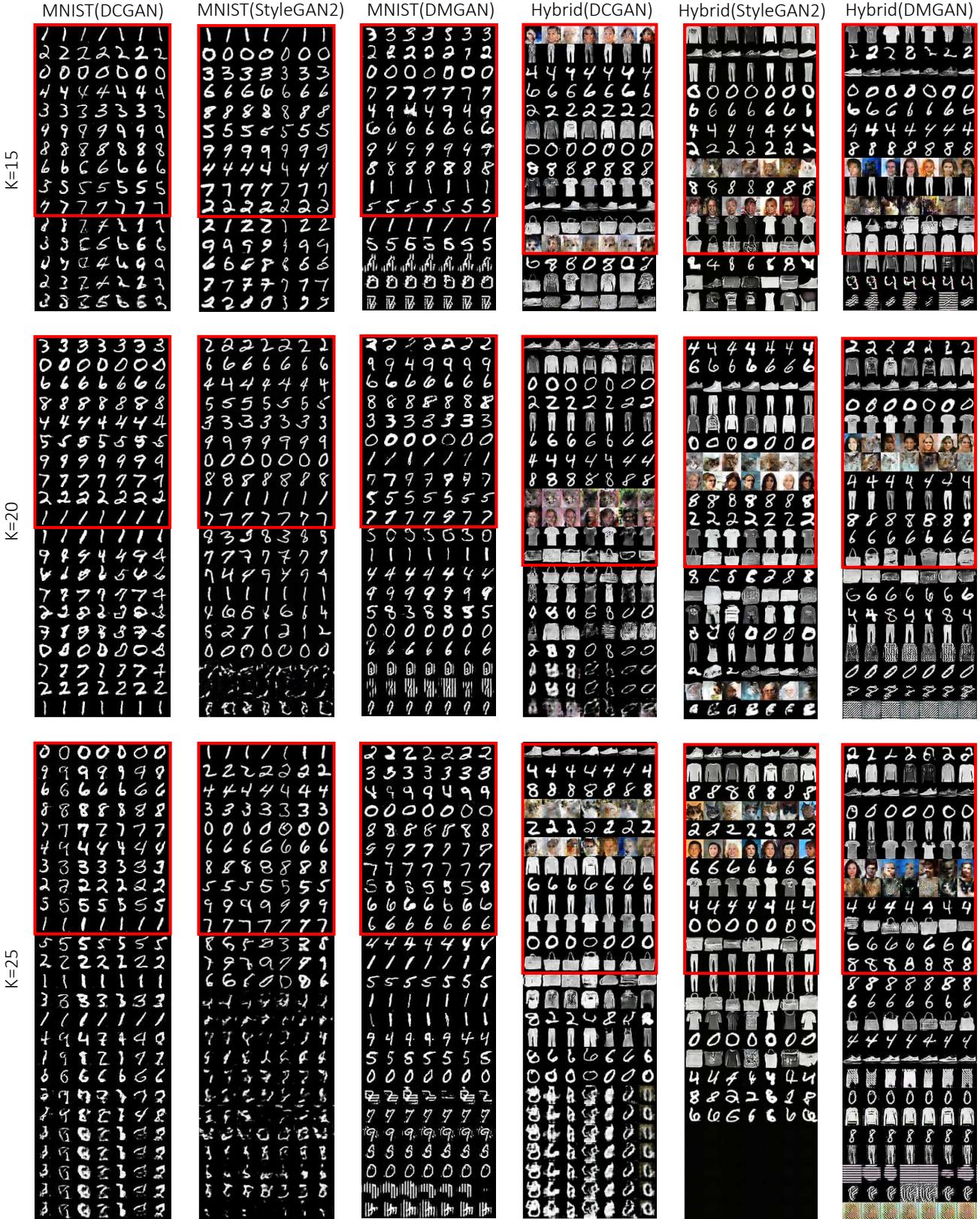
Figure 8. Our results on the MNIST and Hybrid dataset using DCGAN and StyleGAN2 with different Ks, compared to DMGAN. Each row is generated from a mode, and the rows are sorted by αs. The red boxes mark the top n modes in the results, where $n = 10$ for MNIST and $n = 12$ for Hybrid.
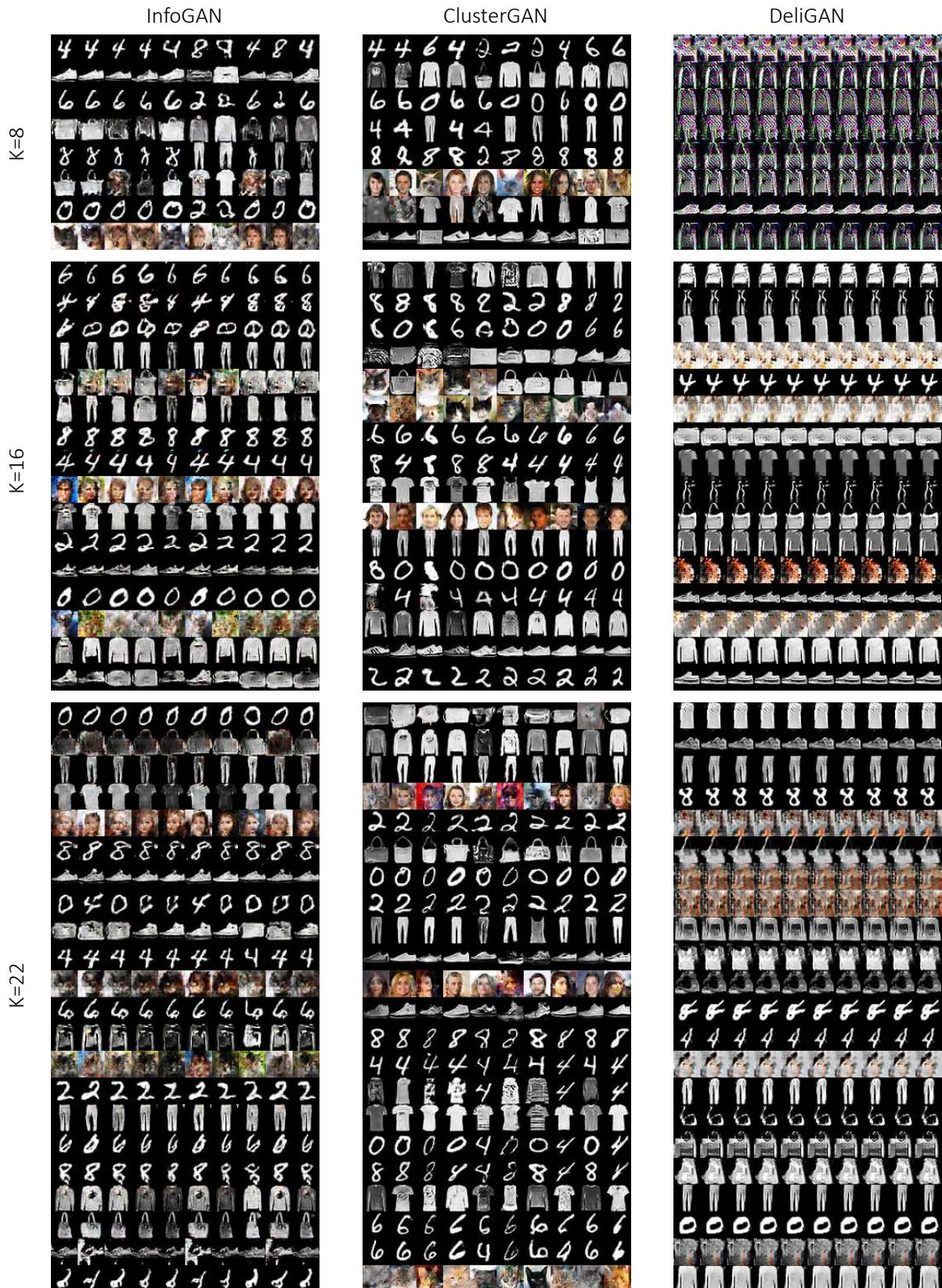
Figure 9. Results of InfoGAN, ClusterGAN and DeliGAN on the Hybrid dataset with different Ks. Each row is generated from a mode.

Figure 10. Results of Our method, Self-Conditioned GAN and StyleGAN2-Ada on the CIFAR with different Cs and Ks. 'C' means the ground-truth class number in the dataset. Note that StyleGAN2-Ada is trained without conditions. For our method and Self-Conditioned GAN, each row is generated from a mode, and the number on the right of each row indicates the distribution of the mode.
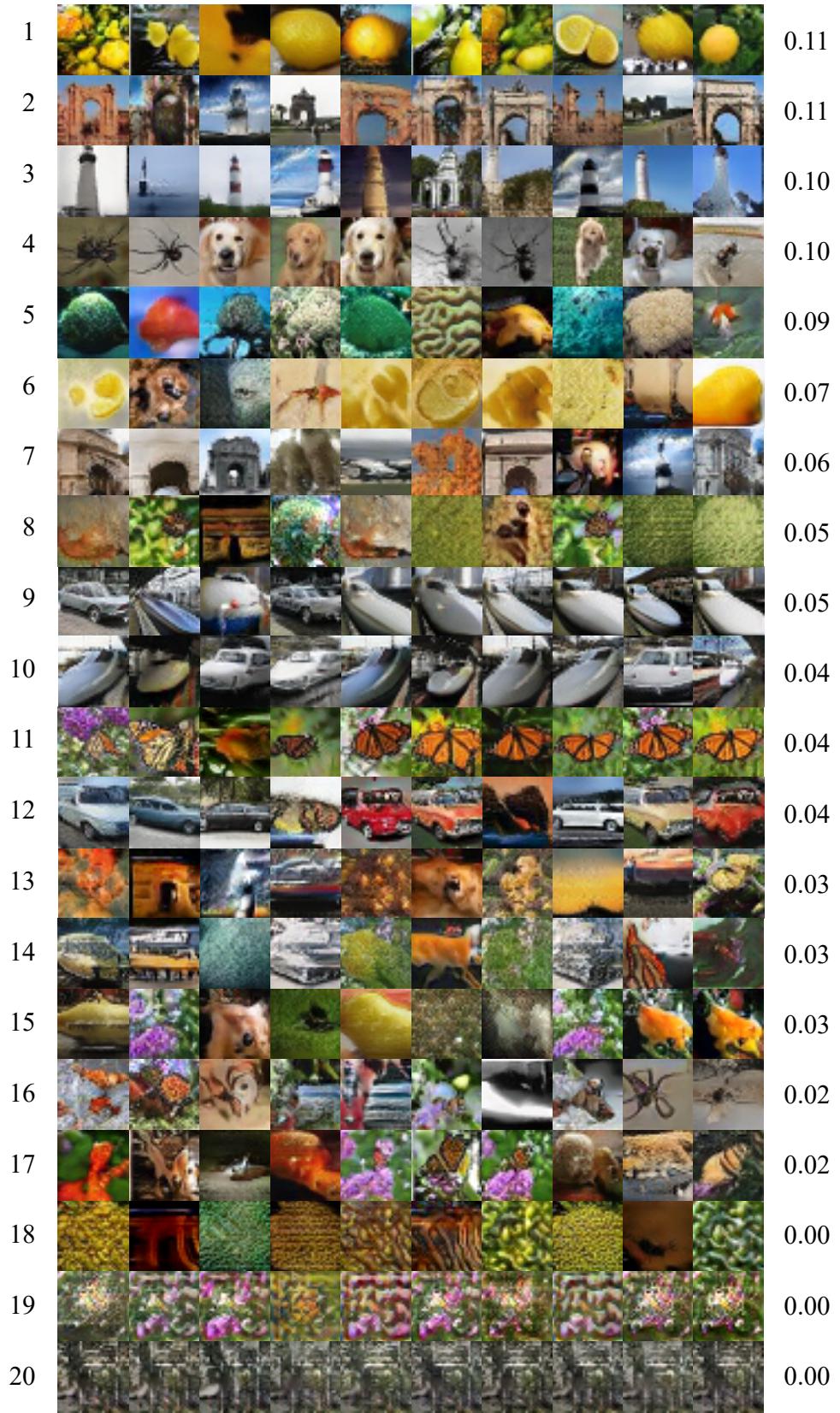
Figure 11. Results of Our method on the Tiny Imagenet with different $C = 10s$ and $K = 20s$. Each row is generated from a mode,. The number on the left of each row indicates the line number and on the right indicates the weight of the mode.