INVERTGAN: REDUCING MODE COLLAPSE WITH MULTI-DIMENSIONAL GAUSSIAN INVERSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative adversarial networks have shown their ability in capturing high-dimensional complex distributions and generating realistic data samples e.g. images. However, existing models still have difficulty in handling multi-modal outputs, which are often susceptible to mode collapse in the sense that the generator can only map latent variables to a part of modes of the target distribution. In this paper, we analyze the typical cases of mode collapse and define the concept of mode completeness in the view of probability measure. We further prove that the inverse mapping can play an effective role to mitigate mode collapse. Under this framework, we further adopt the multi-dimensional Gaussian loss instead of one-dimensional one that has been widely used in existing work, to generate diverse images. Our experiments on synthetic data as well as real-word images show the superiority of our model. Source code will be released with the final paper.

1 INTRODUCTION

For generative models, the real-word data is often assumed to sample from an unknown and implicit distribution. Deep generative models often try to construct a mapping from a known distribution e.g. Gaussian distribution to the implicit target[1]. Recent efforts mainly focus on Variational Auto-Encoders (VAEs) (Kingma & Welling, 2014), Generative Flow models (Rezende & Mohamed, 2015) and Generative Adversarial Networks (GANs) (Goodfellow et al., 2014).

However, many problems still exist for generative models. VAEs embed a given data sample to a hidden Gaussian distribution as representation, which may not be essentially effective for high-quality data generation. Flow models can obtain a one-to-one mapping, at the cost of high computational overhead. GAN can generate high-resolution images, but it often suffers from mode collapse, which tends to only focus on a subset of modes while excludes other parts of the target distribution (Liu et al., 2020), leading to a poor diversity of generated samples.

This paper focuses on solving the mode collapse for GAN. Recent studies address mode collapse mainly in two ways: i) modifying the model learning to achieve a better convergence (Gulrajani et al., 2017; Metz et al., 2017); ii) encouraging the models to learn diverse modes with variance (Elfeki et al., 2019; Meulemeester et al., 2020) or mapping back to learn the representation together (Srivastava et al., 2017; Donahue et al., 2017). The former mainly focuses on the study of convergence between the generated distribution and the target one. While the later tries to encourage the generator to sample fake data with diversity similar to real data.

In this paper, we departure from the popular view centering around the relation between the generated distribution and the real one, but pay more attention to the relation between the source distribution and the target distribution. Specifically, we design an inverse mapping based method to tame mode collapse. Different from previous auto-encoding works with the assumption of one dimensional Gaussian distribution, we regard the inverse of real data with a multi-dimensional Gaussian sampling view and the resulting method is termed as InvertGAN. **Our main contributions include:**

1) We carefully analyze the typical cases of mode collapse, i.e mode missing, mode imbalance, and a mixture of the former two cases. Based on this mapping perspective, we define *mode completeness* in Definition 1, that is, the generative mapping T shall meet the condition of Eq. 1. Then based on

[1]Such a mapping is formally termed by *generative mapping* in this paper.

this definition, in Proposition 1 we prove that the inverse of a real dataset as the target can be viewed as independent samples of the source distribution.

2) Based on our analysis, we propose InvertGAN to address mode collapse. It assumes a multi-dimensional Gaussian distribution based on the inverse of real data and promotes the multi-dimensional distribution to be close to standard Gaussian distribution, which emphasizes the independence between different dimensions for source distribution. Fig. 3 shows that the adoption of multi-dimensional Gaussian, that is, the use of the covariance matrix instead of variance, is better than the traditional single-dimensional method.

3) On synthetic data with multiple modes, we empirically show that InvertGAN outperforms peer methods by different metrics including covered mode number, Quality, reverse KL divergence. On real world images, our simple technique also performs competitively by Inception Score, FID especially for those with high resolution (STL-10) and a large number of categories (CIFAR-100).

2 Related Work

Since its debut (Goodfellow et al., 2014) as an effective generative model, many subsequent works have been proposed to improve the stability and quality of generation. However, GANs still suffer from unstable training, and mode collapse has been one of the most common issues for GANs training. Recent approaches tackle mode collapse mainly in the following different ways:

Improving the training behavior. The Unrolled GAN Metz et al. (2017) presents a surrogate objective to train the generator along with the unrolled optimization of the discriminator, which shows improvements in terms of training stability and reduction of mode-collapse. As an improvement to Wasserstein GANs (Arjovsky et al., 2017), WGAN-GP (Gulrajani et al., 2017) devises a gradient penalty whose effectiveness has shown in the realistic generation system (Karras et al., 2018). (Mescheder et al., 2018) gives the ODE view and proves that zero gradient penalty can improve the convergence for the generation.

Enforcing to capture diverse modes. To generate data with the same diversity as the real data is, many methods are proposed to solve mode collapse for the GAN model. (Elfeki et al., 2019) uses the theory of determinantal point processes and gives a penalty in the layer of the discriminator to enforce the generated data having similar covariance of real data. The approach in (Meulemeester et al., 2020) uses Bure metric instead and discusses more training details. The VEEGAN (Srivastava et al., 2017) design an inverse of the generator and encourage the discriminator to distinguish the joint distribution for real and generated one, which is similar to the work (Donahue et al., 2017).

Multiple generators and discriminators. One direct way of reducing mode-collapse is involving more than one generators to achieve wider coverage for the true distribution. In (Liu & Tuzel, 2016), two coupled generator networks are trained with parameter sharing to jointly learn the true distribution. The multi-agent based system MAD-GAN (Ghosh et al., 2018) involves multiple generators along with one discriminator. The system implicitly encourages each generator to learn their own mode. On the other hand, multiple discriminators are used in (Durugkar et al., 2017) as ensemble. Similarly two additional discriminators are trained to improve the diversity (Nguyen et al., 2017).

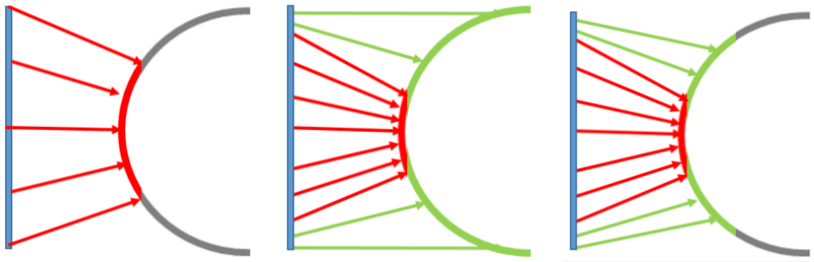
We note these works are orthogonal to our contribution and mostly could be fulfilled in conjunction with ours to further improve the training stability.

3 Inverse Mapping based Probabilistic View to Mode collapse

Despite their success, GANs still suffer the mode collapse issue, especially for complicated distribution. This section presents our inverse mapping based techniques to address this challenge.

3.1 Case Analysis for Mode Collapse

We use Fig. 1 to illustrate typical cases of mode collapse from the perspective of mapping between two probability measures: the left blue line is the known source probability measure α defined in the domain $\mathcal{A}$, while the semicircle on the right is the target β in the domain $\mathcal{B}$. The goal of a generative model is to establish the mapping T from the source distribution to the target one i.e. real data.



(a) Mode missing (b) Mode unbalancing (c) Mixture of the two

Figure 1: Mode collapse in the view of mapping: the left is the source distribution and the right is target. (a) mode missing in grey; (b) mode imbalance as the green part is less mapped from source.

i) Mode missing. Not all the modes can be completely generated i.e. mapped from the source distribution. As shown in Fig. 1(a), almost half part of the probability measure β can not be generated from α, which we call mode missing. In this case, the generative mapping T is not a surjection (i.e. $T(A) \neq B$), which is the main reason for mode missing.

ii) Mode imbalance. As shown in Fig. 1(b), most of the target distribution can be covered yet there exists an imbalance that some part is densely mapped (red) while the rest (green) is more sparse.

iii) Mixture of the two. Mode missing and imbalance are mixed in Fig. 1(c), which will be addressed separately by our techniques, though the former can be treated as a special case of the latter.

Definition 1 (mode completeness for mapping) *For any measurable set $\mathcal{C} \subset \mathcal{B}$ from the target domain, the generative mapping T is defined to be mode complete w.r.t. probability measure from α to β, if T satisfies:*

$$\beta(\mathcal{C}) = \alpha(z \in \mathcal{A} : T(\mathbf{z}) \in \mathcal{C}\}) \tag{1}$$

which is often written as the push-forward operator (Peyré & Cuturi, 2018) $\beta = T_{\#}\alpha$.

Equivalently, the above proposition can be rewritten from a sampling perspective. Given the independent samples $Z = \{\mathbf{z}_i\}_{i=1}^N$ and $X = \{\mathbf{x}_i\}_{i=1}^N$ where $\mathbf{z}_i$ is sampled from α and $\mathbf{x}_i$ is sampled from β. Here N is a large enough number and then we have:

$$\mathbf{card}(X \cap \mathcal{C}) = \mathbf{card}(\{\mathbf{z} \in Z : T(\mathbf{z}) \in \mathcal{C}\}) \tag{2}$$

where $\mathbf{card}(\cdot)$ is the cardinal number of the set. Obviously, there exist infinite solutions for the mapping T and the core of the mode collapse problem is that $T_{\#}\alpha$ does not coincide with the probability measure β of the real data. Therefore, when training T to find the hidden probability measure β, it must be noted that the real data $\{\mathbf{x}_i\}$ is independently sampled from β.

Besides, Eq. 1 has met the condition that T is a surjection. Given a non-empty set $\mathcal{C} \subset \mathcal{B}$ where $\beta(\mathcal{C}) \neq 0$, if $\alpha(\{\mathbf{x} \in A : T(\mathbf{z}) \in \mathcal{C}\}) = 0$ which means $\mathcal{C}$ is the set that can not be mapped from $\mathcal{A}$, we can get $\beta(\mathcal{C}) = 0$. It contradicts the precondition $\beta(\mathcal{C}) \neq 0$. For mode imbalance in Fig. 1(b), the subset $\mathcal{C}$ and $\{\mathbf{x} \in A : T(\mathbf{z}) \in \mathcal{C}\}$ share the same probability, which prevents the imbalance for the number of independent samples in the corresponding domain.

3.2 An Inverse Method for Mode Collapse

Inverse methods have been applied (Srivastava et al., 2017) in GANs to improve the generation and translation. Here we will show that the inverse method can be used to address mode missing and imbalance. Previous works (Kingma & Welling, 2014; Srivastava et al., 2017) mainly

We turn to designing an inverse mapping T^{-1} to address mode missing, which can fulfill surjection of mapping T. More specifically, for any target sample $\mathbf{x}$, the constrain $T^{-1}(\mathbf{x}) \in A$ can be enforced for training T and its inverse T^{-1}, which ensures the existence of the corresponding $\mathbf{z}$ for $T(\mathbf{z}) = \mathbf{x}$.

Proposition 1 (inverse constraints for target samples) *If the generative mapping T is mode complete from probability measure α to β and its inverse T^{-1} exists, and if $\{\mathbf{x}_i\}_{i=1}^n$ are n independent samples from β, then $\{\tilde{\mathbf{z}} = T^{-1}(\mathbf{x}_i)\}_{i=1}^n$ can be viewed as n independent samples from β.*

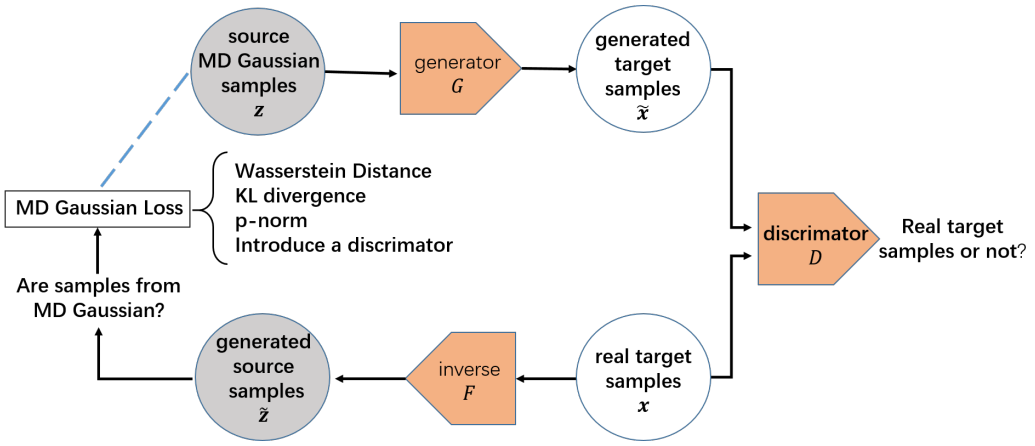


Figure 2: The InvertGAN: generator G maps random samples from source standard MD Gaussian to target ones and F inverts the target sample back to a source sample obeying MD Gassuian.

Proof If the mapping T is mode complete according to Definition 1, then Eq. 1 is satisfied. Given the inverse T^{-1}, Eq. 1 can be transformed into:

$$\beta(\mathcal{C}) = \alpha\{\tilde{\mathbf{z}} \in \mathcal{A} : \tilde{\mathbf{z}} = T^{-1}(\mathbf{x}_i), \mathbf{x}_i \in \mathcal{C}\} \tag{3}$$

So by setting $\mathcal{C} = \{x_1\}, \{x_2\}, \ldots, \{x_n\}$ respectively, we can find that the corresponding probabilities for $\{\mathbf{x}_i\}$ and $\{\tilde{\mathbf{z}}_i\}$ are equal. It means that $\{T^{-1}(\mathbf{x}_i) : \mathbf{x}_i \in \mathcal{C}\}$ can be regarded as independent samples from α.

Therefore, given the real data $\{\mathbf{x}_i\}_{i=1}^n$ which is often viewed as independent samples from some unknown distribution, we only need to train the inverse mapping so that $\{T^{-1}(\mathbf{x}_i)\}$ is independently sampled from the probability measure α, and we can get diverse samples by independent sampling.

4 The Proposed Model

The original GAN model consists of a discriminator $D : R^d \to R$ and a generator $G : R^l \to R^d$, which are typically embodied by deep neural networks. Given the empirical distribution $P_{\mathbf{x}}$, $D(\mathbf{x})$ as used to distinguish generator samples from true data samples, while $G(\mathbf{z})$ is the mapping from Gaussian sample $\mathbf{z}$ to a point in the data space R^d. The discriminator and generator are optimized by solving the minimax problem, by alternating the two phases of training:

$$\min_G \max_D V(G, D) = E_{x \sim P_x}\left[\log(D(\mathbf{x}))\right] + E_{z \sim P_z}\left[\log(1 - D(G(\mathbf{z})))\right] \tag{4}$$

The first term gives the expectation of probability that $\mathbf{x}$ comes from real data distribution $P_{\mathbf{x}}$ and the second involves an input distribution P_z, which is embodied by a standard multi-dimensional Gaussian distribution $\mathcal{N}(0, I_l)$ in this paper. Here l is the dimension of $\mathbf{z}$. Later in this paper we will elaborate the reason why a multi-dimensional Gaussian is used which differs from existing works using a single-dimensional Gaussian.

4.1 Inverse Mapping for Mode Missing

Recall that in Sec. 3.2, we have shown that designing the inverse mapping can be used to reduce mode collapse. We adopt the neural network F as the inverse of the generator. To achieve $F = G^{-1}$ and solve the mode missing problem, we design the follow loss:

$$L_{cons}(G, F) = \underbrace{E_{z \sim P_z}\|z - F(G(\mathbf{z}))\|_2}_{\text{to achieve inverse mapping}} + \underbrace{E_{x \sim P_x}\|x - G(F(\mathbf{x}))\|_2}_{\text{to avoid mode missing}} \tag{5}$$

The first term promotes F be the inverse of G, which uses the reconstruction penalty as an expectation of the cost of autoencoding noise vectors (Srivastava et al., 2017) and the second term promotes that $F(\mathbf{x}) \in R^l$, which makes $\tilde{\mathbf{z}}$ exist in R^l. Then for every real data point $\mathbf{x}$, we can find the corresponding $\tilde{\mathbf{z}}$ in R^l which satisfies $G(\tilde{\mathbf{z}}) = \mathbf{x}$.

4.2 MD GAUSSIAN LOSS FOR MODE CONCENTRATION

As mentioned in Sec. 3.2, given the real data $\{\mathbf{x}_i\}$ sampled from P_x, $\{\tilde{\mathbf{z}} = F(\mathbf{x}_i)\}$ follows the distribution P_z, which can be viewed as i.i.d samples from P_z. So to solve mode imbalance, the inverse mapping F should make $\{\tilde{\mathbf{z}}\}$ more like being sampled from standard Gaussian P_z.

We propose to use multi-dimensional Gaussian (**MD Gaussian**) as the source distribution instead of the widely used 1D Gaussian (Srivastava et al., 2017). The reasons are two folds:

1) The source 1D Gaussian used in previous works can be regarded as a standard MD Gaussian whose dimensions are independent to each other (in the sense that each sampling from a 1D Gaussian can be regarded as sampled from a standard MD Gaussian along a certain dimension). However, the inverse samples $\{\tilde{z}\}$ are always dependent if we use 1D Gaussian; 2) There is a correlation among the data points, and using covariance instead of variance can better capture this correlation with enhanced model capacity.

So we assume $\tilde{\mathbf{z}}_i = T(\mathbf{x}_i)$ follow the MD Gaussian $P_{\tilde{z}}$ with its mean $\boldsymbol{\mu}_{\tilde{z}}$ and variance $\boldsymbol{\Sigma}_{\tilde{z}}$, then our goal is to make $P_{\tilde{z}}$ approximate to $\tilde{\mathbf{z}}_i = T(\mathbf{x}_i)$ by training, so that the real data can be considered as conversion of samples from P_z and the mode will not collapse. To get the approximation, we can design the Gaussian loss L_{Gau} for the inverse F with the following distances or divergence:

1) Wasserstein distance. The Wasserstein distance has been widely used to evaluate the distance between two distributions. Given two MD Gaussains P_z and $P_{\tilde{z}}$, the 2-Wasserstein distance is:

$$W_2(P_z, P_{\tilde{z}}) = \sqrt{\|\boldsymbol{\mu}_{\tilde{z}}\|^2 + trace(\boldsymbol{\Sigma}_{\tilde{z}} + I_l - 2\boldsymbol{\Sigma}_{\tilde{z}}^{1/2})} \tag{6}$$

We can see that $W_2(P_z, P_{\tilde{z}}) = 0$ if and only if $\boldsymbol{\mu}_{\tilde{z}} = 0$ and $\boldsymbol{\Sigma}_{\tilde{z}} = I_l$.

2) KL divergence. It is an important divergence to measure the difference between two distributions. Given MD Gaussians P_z and $P_{\tilde{z}}$, the KL divergence $KL(P_z, P_{\tilde{z}})$ can be specified as:

$$KL(P_z, P_{\tilde{x}}) = \frac{1}{2}\left\{\log(\det(\boldsymbol{\Sigma}_{\tilde{z}})) - l + trace(\boldsymbol{\Sigma}_{\tilde{z}}^{-1}) + \boldsymbol{\mu}_{\tilde{z}}{}^{\top}\boldsymbol{\Sigma}_{\tilde{z}}{}^{-1}\boldsymbol{\mu}_{\tilde{z}}\right\} \tag{7}$$

3) p-norm distance. In essence, the Wasserstein distance and KL divergence use the trace of matrix, which may pay more attention to the difference of its eigenvalues. Here we define a simple distance between MD Gaussian distance called p-norm distance. Given two Gaussians $\gamma_a = \mathcal{N}(\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a)$ and $\gamma_b = \mathcal{N}(\boldsymbol{\mu}_b, \boldsymbol{\Sigma}_b)$, we specify the p-norm distance $D^p(\gamma_a, \gamma_b)$ as:

$$D^p(\gamma_a, \gamma_b) = \|\boldsymbol{\mu}_a - \boldsymbol{\mu}_b\|_p + \|\boldsymbol{\Sigma}_a - \boldsymbol{\Sigma}_b\|_p \tag{8}$$

where $\|\cdot\|_p$ is the p-norm for the vector. For matrix, we set $\|\Sigma\|_p = (\sum_i \sum_j \Sigma_{ij}^p)^{\frac{1}{p}}$. We can prove that L^p is a distance because $\|\cdot\|_p$ is a norm and satisfies the the triangle inequality as given by:

$$D^p(P_z, P_{\tilde{z}}) = \|\boldsymbol{\mu}_{\tilde{z}}\|_p + \|\boldsymbol{\Sigma}_{\tilde{z}} - I_l\|_p \tag{9}$$

4) The $\tilde{\mathbf{z}}-$discriminator. It discriminates whether the generated image is a sample of real distribution $P_{\mathbf{x}}$. Similarly, we can also use the discriminator to distinguish the difference between real Gaussian samples and generated ones. By inputting as many samples sampled from P_z as possible, together with the corresponding inverse samples $\{\tilde{\mathbf{z}}\}$ from the real data, we optimize D_z and F by adversarial learning to push $\{\tilde{\mathbf{z}}\}$ closer to the sampling results of P_z. Then we can get our final loss as:

$$\min_{G,F}\max_{D} V(G, D) + L_{cons}(G, F) + L_{Gau}(F) \tag{10}$$

where $L_{Gau}(F)$ can be specified based on different distances or divergence (the method with $\tilde{\mathbf{z}}-$discriminator should train one more discriminator. We will discuss its final loss in Appendix).

1D Gaussian Loss. To show the superiority of MD Gaussian loss, here we also test the 1D Gaussian loss which is given as follows, where $\boldsymbol{\sigma}_{\tilde{z}}$ is the standard deviation of $\{\tilde{\mathbf{z}}\}$.

$$W_2(P_z, P_{\tilde{z}}) = \sqrt{\|\boldsymbol{\mu}_{\tilde{z}}\|^2 + \|\boldsymbol{\sigma}_{\tilde{z}} - 1\|^2} \tag{11}$$

Fig. 3 and Fig. 5 show the superiority of MD Gaussian loss which will be detailed in experiments.

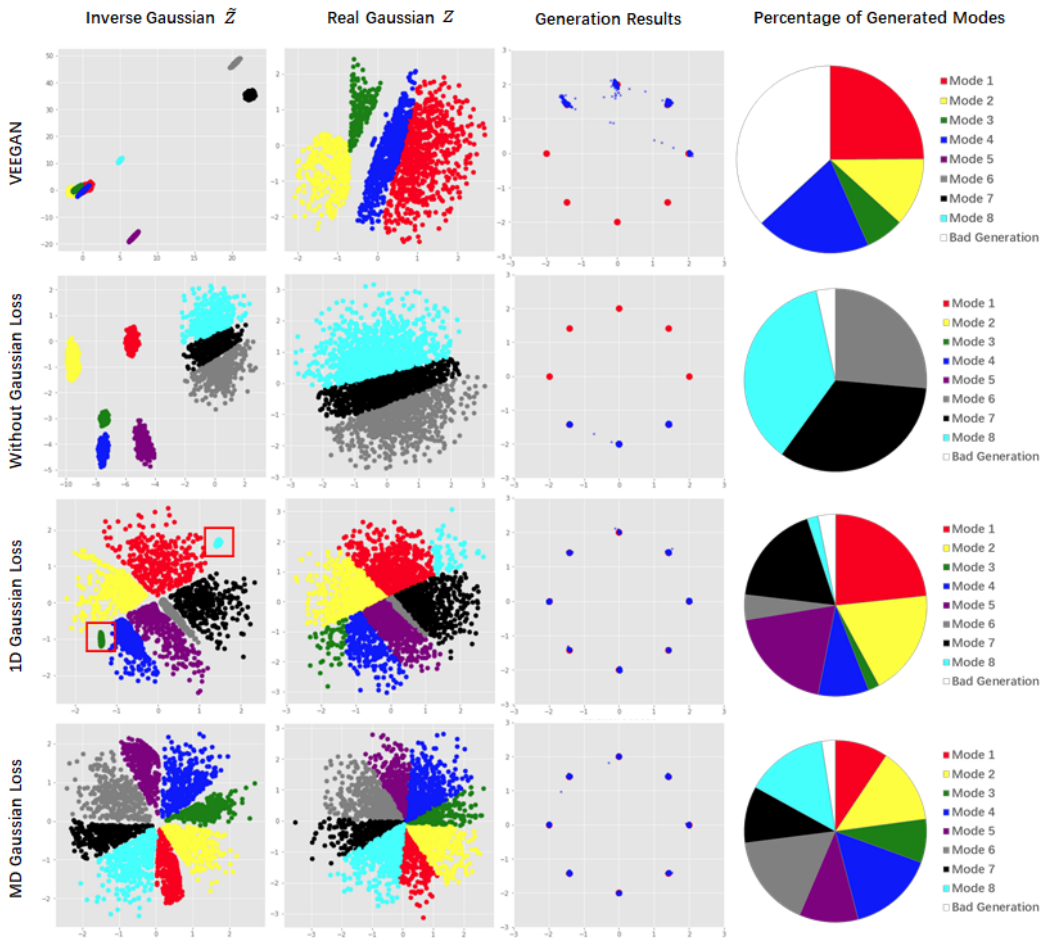


Figure 3: Results visualization for Ring, by comparing VEEGAN and InvertGAN under different Gaussian losses as detailed in Sec. 4.2 after training 24K mini-batches. Both 1D and MD Wasserstein distance are used. The first column is the inverse $\{\tilde{\mathbf{z}}_i\}$ of real data $\{\mathbf{x}_i\}$, which has the same number of points for each color (i.e. mode). The red box in the third row represents that 1D Gaussian loss may cause some modes to gather and affect the balance. The second column refers to random sampling based on standard Gaussian P_z, which maps to different modes. The third column is the random generation result, and the fourth column is the generation percentage of each mode.

5 Experiments

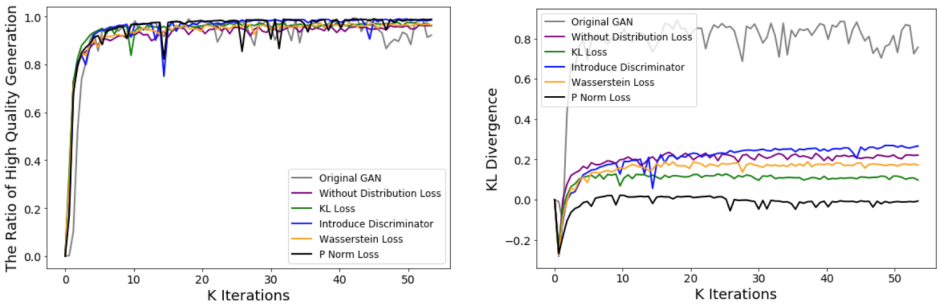
The experiments cover both synthetic dataset and real-world images. To highlight the advantages and disadvantages of the model in design, we adopt a simpler network architecture for both synthetic and real world experiments to more directly evaluate the contributions of our techniques.

5.1 Experiments on Synthetic Datasets

Mode collapse can be accurately measured on synthetic data, as the real distribution is known. In line with (Metz et al., 2017), we simulate two synthetic datasets. The batch size is set to 128.

Ring: a mixture of eight 2D Gaussians with their mean $\{(2\cos(i\pi/4), 2\cos(i\pi/4))\}_{i=1}^{8}$ and the standard deviation 0.001. 12.5K samples are simulated from each Gaussian distribution i.e. 100K samples in total. 50K samples from P_z are used to generate $\mathbf{x}$ for testing.

Grid: a mixture of 25 2D isotropic Gaussians with mean $\{(2i, 2j)\}_{i,j=-2}^{2}$ and standard deviation 0.0025. 4K samples are simulated from each Gaussian (i.e. 100K samples in total). 100K samples from P_z are used to generate target samples $\{\tilde{x}\}$ for testing.



(a) Generation quality (b) Reverse KL divergence

Figure 4: Comparison of different Gaussian distribution loss on the Ring synthetic data.

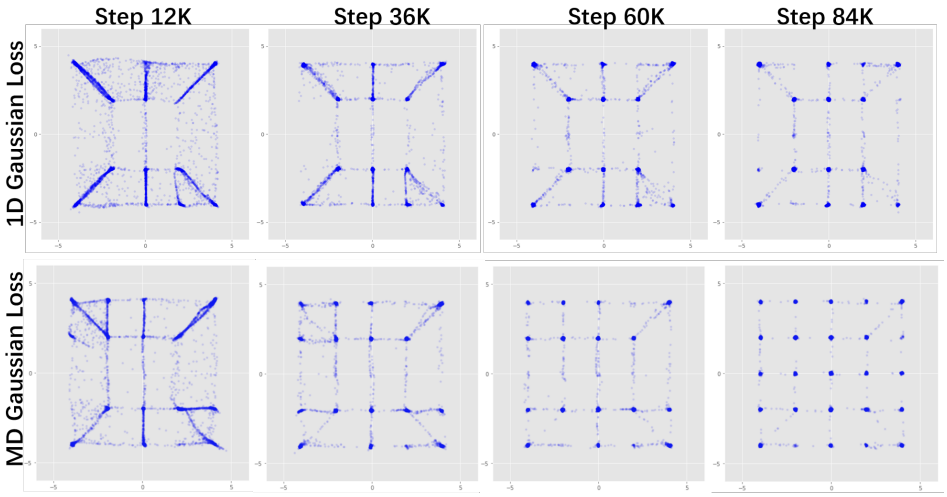


Figure 5: Generations of grid data given poor initialization. Compared with 1D Gaussian loss, MD Gaussian loss can overcome the mode collapse after enough training steps (i.e. batch iteration). The comparison on Ring is given in appendix which similarly shows the advantages of our methods.

For the network architecture, (Metz et al., 2017) suggest that the activation function tanh can improve training. Different from (Metz et al., 2017; Elfeki et al., 2019), we are refrained from this trick to more directly verify the role of our techniques. So in the synthetic experiment, we use four linear layers with ReLu activation function for testing.

For metrics, following (Metz et al., 2017; Elfeki et al., 2019), we use the numbers of modes covered, generation quality[2] and reverse KL divergence. Since in the experiment, each mode shares the same number of real samples, it can be used to calculate the reverse KL divergence between the generated distribution and the real one (Nguyen et al., 2017): $D_{KL}(model||data) = \sum_{i=1}^{m} p_i \log \frac{p_i}{1/m}$.

Note the reverse KL divergence is not strictly defined because $\sum_{i=1}^{m} p_i < 1$ (i.e. there exist poor generated points). So the reverse KL divergence allows to be negative as shown in Fig. 4(b).

Superiority of MD Gaussian loss. As shown in Fig. 3, MD Gaussian loss gives an good result on synthetic data. Compared with methods without using Gaussian loss and VEEGAN, InvertGAN with Gaussian loss does not view the data in isolation, but considers them as a whole for the entire batch. It is clear that InvertGAN with Gaussian loss performs well as shown in Fig. 3 that every mode can be recovered and the inverse of real samples are close to the center of standard Gaussian.

Compared with 1D Gaussian, MD Gaussian loss makes the inverse of real data more close to real Gaussian samples. As shown in the third row in Fig. 3, 1D Gaussian loss can easily make it imbalance and MD Gaussian loss overcomes this limitation. Besides, it is well known that the training of

[2]The definition is (Meulemeester et al., 2020): if the generated data point is within 3 times standard deviation of the Gaussian mean, consider it a good generated point and the resulting ratio is used as the generation quality.

Table 1: Comparison results on the synthetic data: Ring and Grid (best in bold).

Models	2D-Ring			2D-Grid		
	Mode Score↑	Quality% ↑	RKL↓	Mode↑	Quality% ↑	RKL↓
GAN	3.6 ± 0.5	98.8 ± 0.6	0.92 ± 0.11	18.4 ± 1.6	$\mathbf{98.0 \pm 0.4}$	0.75 ± 0.25
BiGAN	6.8 ± 1.0	38.6 ± 9.5	0.43 ± 0.18	24.2 ± 1.2	83.4 ± 2.9	$\mathbf{0.26} \pm 0.20$
Unrolled GAN	6.4 ± 2.2	98.6 ± 0.5	0.42 ± 0.53	8.2 ± 1.7	98.7 ± 0.6	1.27 ± 0.17
VEEGAN	5.4 ± 1.2	38.8 ± 16.7	0.40 ± 0.10	20.0 ± 2.6	85.0 ± 5.9	0.41 ± 0.10
InvertGAN (ours)	$\mathbf{8.0 \pm 0.0}$	$\mathbf{99.0 \pm 0.2}$	$\mathbf{0.17} \pm 0.06$	$\mathbf{25.0 \pm 0.0}$	$\mathbf{98.0 \pm 0.4}$	$\mathbf{0.26 \pm 0.12}$

Table 2: Comparison results on real-world datasets: CIFAR-10 and CIFAR-100 (best in bold).

Models	CIFAR10			CIFAR100			STL-10		
	IS↑	FID↓	MS↑	IS↑	FID↓	MS↑	IS↑	FID↓	MS↑
GAN	1.88	262.91	1.86	5.03	74.41	5.09	2.28	245.21	2.29
BiGAN	3.30	184.37	3.27	4.41	75.27	4.49	1.22	251.21	1.22
Unrolled GAN	**4.72**	**74.05**	**4.67**	**6.15**	79.14	**6.14**	4.78	142.16	4.62
VEEGAN	3.71	120.62	3.71	4.80	75.86	4.85	1.45	298.95	1.46
InvertGAN (ours)	4.68	75.80	**4.67**	5.33	**74.26**	5.30	**5.16**	**139.10**	**5.12**

GANs is sometimes sensitive to bad initialization which leads to mode missing as shown in Fig .5. However, MD Gaussian loss performs more robustly.

Comparing different MD Gaussian loss. Fig. 4 shows the results of InvertGAN with different MD Gaussian losses. Fig.4(a) shows the quality of generation results. It is clear that all the methods are approaching 100%. However, in Fig. 4(b), we can find the difference and p-norm loss gets a best result. Thus we use p-norm loss as the MD Gaussian loss to compare with other methods in Table 1.

Comparing different methods. Our InvertGAN is compared with vanilla GAN (Goodfellow et al., 2014), BiGAN (Donahue et al., 2017), Unrolled GAN (Metz et al., 2017) and VEEGAN (Srivastava et al., 2017) on Ring and Grid synthetic datasets. It can be found that our InvertGAN obtains the best and stable performance as shown in Table 1 with significant improvement.

5.2 Experiments on Real-world Dataset

The experiments are performed on three datasets including CIFAR-10, CIFAR-100, and STL-10. Compared with CIFAR-10, images in STL-10 are of higher resolution. All compared models are trained by 100K steps (i.e. number of batches whose size is 64). The image quality is assessed according to the Inception Score (Salimans et al., 2016), Fréchet Inception Distance (FID) (Heusel et al., 2017) and Mode Score (Che et al., 2017), whose computing are all mainly based on the Inception network Szegedy et al. (2016). The results are calculated based on 10K generated images for CIFAR-10 and CIFAR-100, and based on 5K generated images for STL-10. The detailed information for network architectures for the real-world images is given in the appendix.

The comparison is given in Table 2 which suggests InvertGAN performs competitively, especially on the challenging STL-10 and CIFAR-100. Here we use the p-norm distance as MD Gaussian loss. Most notably, our InvertGAN has never encountered training failure e.g. gradient explosion which all the other compared methods struggled during the training in our experiment.

More visual results and comparison are given in the appendix.

6 Conclusion and Outlook

In this paper, we have analyzed the typical cases of mode collapse and define the concept of mode completeness from the mapping perspective between two probability measures. Our devised inverse mapping, as well as the multi-dimensional Gaussian loss show their effectiveness to address the mode collapse issue, on both challenging synthetic dataset and real-world images.

The future work will give more in-depth studies on the role of different loss designs as well as combination of other techniques to further reduce mode collapse.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pp. 214–223, 2017.

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

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *ICLR*, 2017.

I. Durugkar, I. Gemp, and S. Mahadevan. Generative multi-adversarial networks. In *ICLR*, 2017.

Mohamed Elfeki, Camille Couprie, Morgane Riviére, and Mohamed Elhoseiny. Gdpp: Learning diverse generations using determinantal point processes. In *ICML*, 2019.

A. Ghosh, V. Kulharia, V. Namboodiri, P. H. Torr, and P. K. Dokania. Multi-agent diverse generative adversarial networks. In *CVPR*, 2018.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *NIPS*, 2017.

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.

T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018.

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

M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *NIPS*, 2016.

Steven Liu, Tongzhou Wang, David Bau, Jun-Yan Zhu, and Antonio Torralba. Diverse image generation via self-conditioned gans. In *CVPR*, 2020.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *ICML*, 2018.

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

Hannes Meulemeester, Joachim Schreurs, Michael Fanuel, Bart Moor, and Johan Suykens. The bures metric for taming mode collapse in generative adversarial networks. In *CVPR*, 2020.

T. Nguyen, T. Le, H. Vu, and D. Phung. Dual discriminator generative adversarial nets. In *NIPS*, 2017.

Gabriel Peyré and Marco Cuturi. Computational optimal transport. 2018.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, 2015.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, 2016.

Akash Srivastava, Lazar Valkov, Chris Russell, Michael Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *NIPS*, 2017.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

A APPENDIX

A.1 FINAL LOSS FOR INTRODUCING A DISCRIMINATOR D_z

We introduce a discriminator to distinguish whether it is from standard Gaussian distribution. We can get the final loss as

$$\min_{G,F} \max_{D,D_z} V(G,D) + L_{cons}(G,F) + L_{Gau}(F,D_z) \tag{12}$$

where $L_{Gau}(F,D_z)$ can be defined as

$$E_{\mathbf{z}\sim P_z}\left[\log(D_z(\mathbf{z}))\right] + E_{x\sim P_x}\left[\log(1-D_z(F(\mathbf{x})))\right] \tag{13}$$

Through alternating training, we can get the optimal G, F and D, D_z.

A.2 NETWORK ARCHITECTURES

Table 3: Network Architecture for Inverse F.

Layer	Output size	Kernel
Conv2d	32×32	$3 \times 3, 64$
ResnetBlock	32×32	$3 \times 3,\ 64$ $3 \times 3,\ 64$
AvgPool2d	16×16	3×3, stride 2
ResnetBlock	16×16	$3 \times 3,\ 64$ $3 \times 3,\ 128$ $1 \times 1,\ 128$
AvgPool2d	8×8	3×3, stride 2
ResnetBlock	8×8	$3 \times 3,\ 128$ $3 \times 3,\ 256$ $1 \times 1,\ 256$
AvgPool2d	4×4	3×3, stride 2
ResnetBlock	4×4	$3 \times 3,\ 256$ $3 \times 3,\ 512$ $1 \times 1,\ 512$
Linear	20	

Table 4: Network Architectures for Generator G.

Layer	Output size	Kernel
Linear	8192	
ResnetBlock	4×4	$3 \times 3,\ 256$ $3 \times 3,\ 256$ $1 \times 1,\ 256$
Upsample	8×8	scale factor = 2.0
ResnetBlock	8×8	$3 \times 3,\ 128$ $3 \times 3,\ 128$ $1 \times 1,\ 128$
Upsample	16×16	scale factor = 2.0
ResnetBlock	16×16	$3 \times 3,\ 64$ $3 \times 3,\ 64$ $1 \times 1,\ 64$
Upsample	32×32	scale factor = 2.0
ResnetBlock	32×32	$3 \times 3,\ 64$ $3 \times 3,\ 64$
Conv2d	32×32	

A.3 MORE RESULTS FOR SYNTHETIC DATA

A.3.1 RING DATA

The generation results for Ring compared with other methods are:

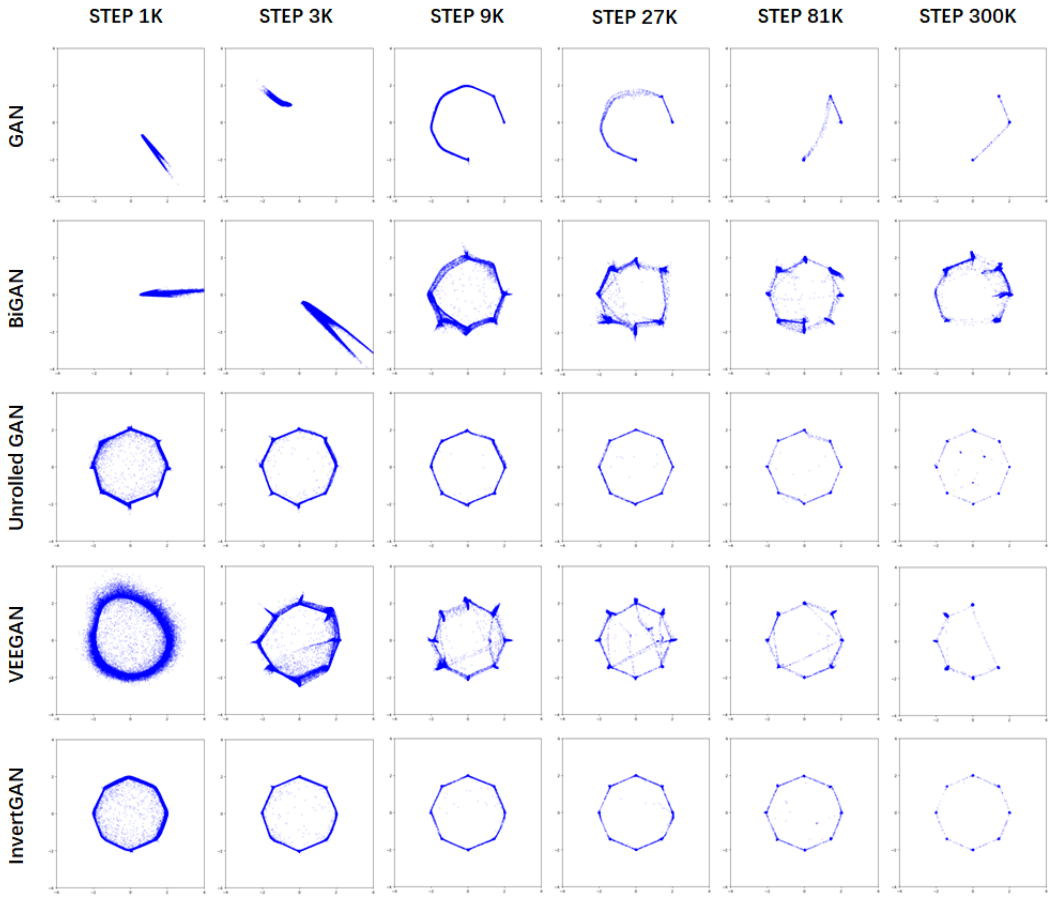


Figure 6: Comparison among different methods for Ring .

A.3.2 GRID DATA

The generation results for Grid compared with other methods are:

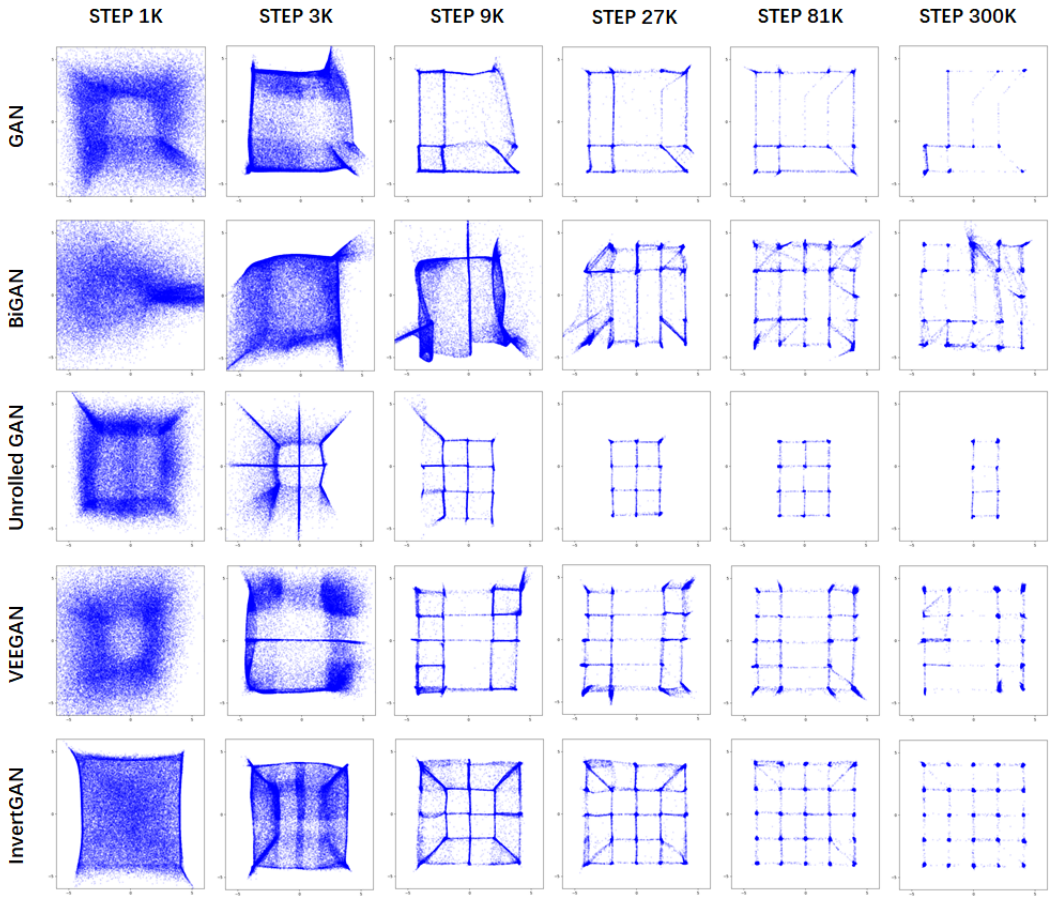
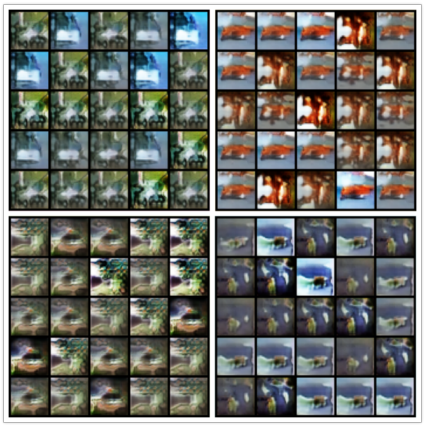


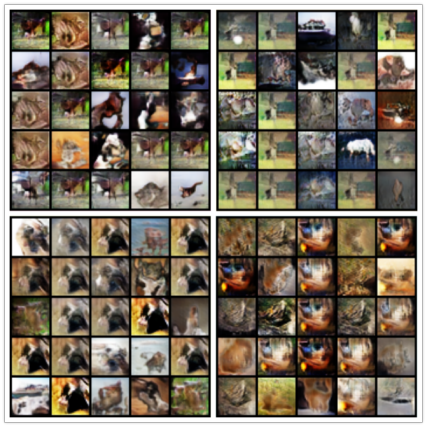
Figure 7: Comparison among different methods for Grid .

A.4 QUALITATIVE RESULTS FOR REAL DATA

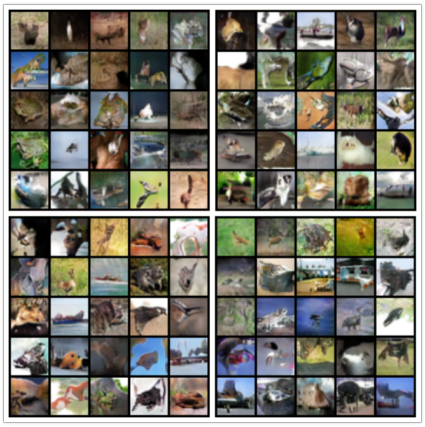
A.4.1 CIFAR10 QUALITATIVE RESULTS



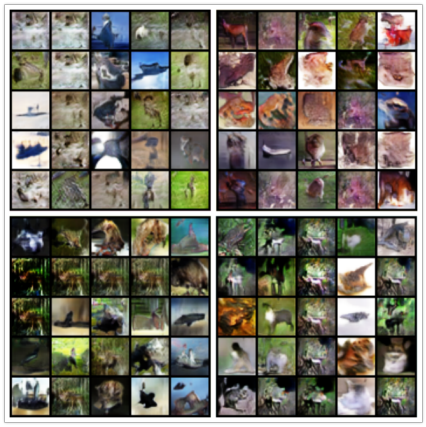
(a) Original GAN



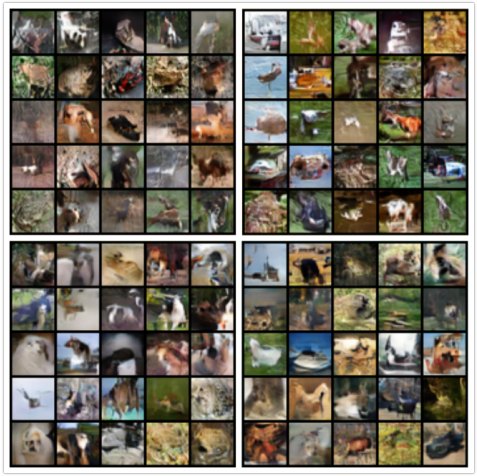
(b) BiGAN



(c) Unrolled GAN



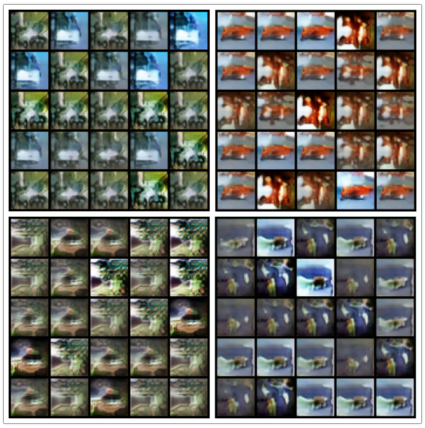
(d) VEEGAN



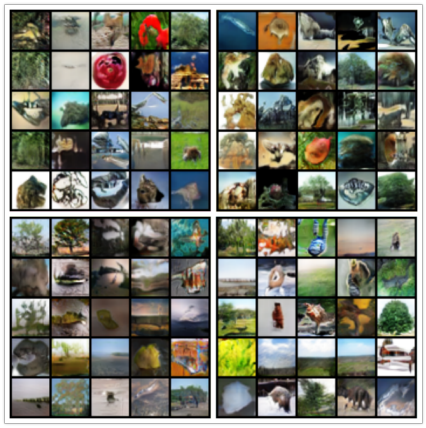
(e) InvertGAN

Figure 8: Comparison among different methods for cifar10 .

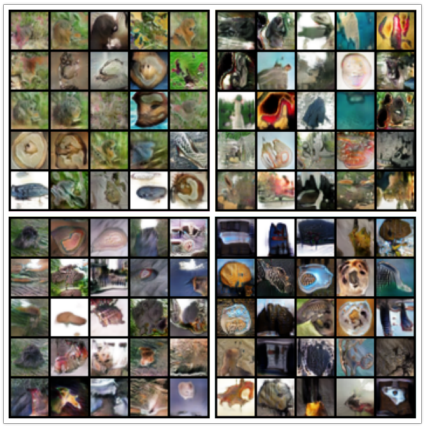
A.4.2 CIFAR100 QUALITATIVE RESULTS



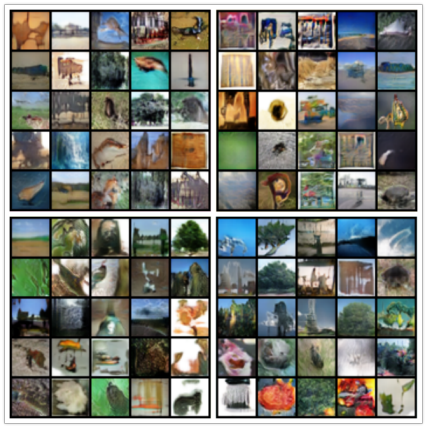
(a) Original GAN



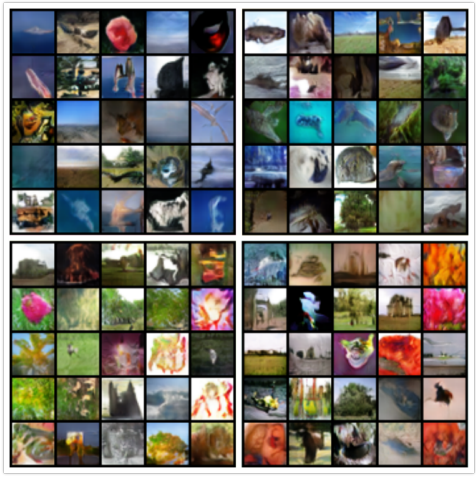
(b) BiGAN



(c) Unrolled GAN



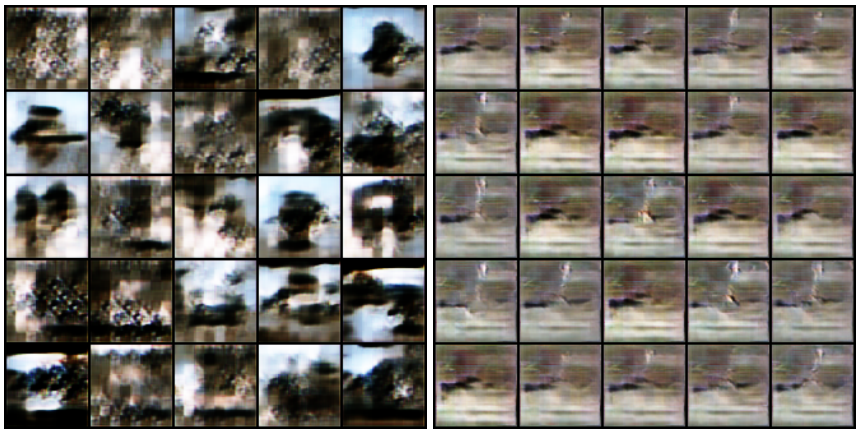
(d) VEEGAN



(e) InvertGAN

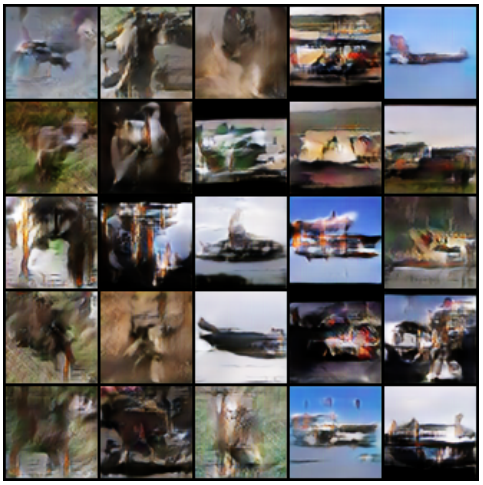
Figure 9: Comparison among different methods for cifar100 .

A.4.3 STL-10 QUALITATIVE RESULTS



(a) Original GAN (b) BiGAN

(c) Unrolled GAN (d) VEEGAN



(e) InvertGAN

Figure 10: Comparison among different methods for STL-10 .