Optimizing Latent Distributions for Non-Adversarial Generative Networks

Tianyu Guo, Chang Xu, Boxin Shi, Senior Member, IEEE, Chao Xu, and Dacheng Tao, Fellow, IEEE

Abstract—The generator in Generative Adversarial Networks (GANs) is driven by a discriminator to produce high-quality images through an adversarial game. At the same time, the difficulty of reaching a stable generator has been increased. This paper focuses on non-adversarial generative networks that are trained in a plain manner without adversarial loss. The given limited number of real images could be insufficient to fully represent the real data distribution. We therefore investigate a set of distributions in a Wasserstein ball centred on the distribution induced by the training data and propose to optimize the generator over this Wasserstein ball. We theoretically discuss the solvability of the newly defined objective function and develop a tractable reformulation to learn the generator. The connections and differences between the proposed non-adversarial generative networks and GANs are analyzed. Experimental results on real-world datasets demonstrate that the proposed algorithm can effectively learn image generators in a non-adversarial approach, and the generated images are of comparable quality with those from GANs.

Index Terms—Non-adversarial generation; image generation; distribution optimization;

1 INTRODUCTION


Generative Adversarial Networks (GANs) demonstrate impressive image generation capabilities. Two neural networks are involved in GANs framework and considered as two players in a game. A discriminator network is trained to distinguish fake samples generated by a generator and those real samples. The generator is trained to generate samples from the noise input and fool the discriminator. These two networks are optimized in turns during the training period. After reaching equilibrium, the generator is expected to generate sufficiently realistic samples that are failed to be distinguished from real samples by the discriminator. An image-to-image translation framework can be easily established by replacing input noises as images, which leads to various applications, e.g., image in-painting [8], super-resolution [9], image denoising, and style transformation [10], [11], [12]. However, vanilla GANs usually suffer from a lack of stability during training. DCGAN [13] introduced a network structure that works well and is stable to alleviate the challenges in training GANs. WGAN [14], WGAN-GP [15], and SNGAN [16] observed the defect of JS-divergence in vanilla GANs and suggested new loss functions to train GANs. Progressive GAN [17] and BigGAN [18] generated high-quality images by improving training methods, e.g., progressively deepening the network, enlarging batch size, and truncating the latent space.

Recently, non-adversarial alternatives take a different way to tackle the disadvantages of GANs for training generative models. VAE [4], WAE [19], and DSD [20] generate images without adversarial loss. As a pioneering attempt in this direction, Generative Latent Optimization (GLO) [21] embeds training images in low dimensional space and reconstructs the images by passing embeddings through a deep generator network. Though without the adversarial training, GLO can still generate visually-appealing images, and more importantly its learned latent space has corresponding semantic image properties. However, its disadvantage is also obvious. GLO puts major efforts to real images and their associated embeddings but lacks a principled way to optimize the latent space for synthesizing new images. It is, therefore, challenging for GLO to generate high-quality images from an arbitrarily given latent code.

In this paper, we propose to optimize a latent Wasserstein ball of distributions for synthesizing images in a non-adversarial manner. Embedding of real images in the training set can only occupy parts of the latent space. Instead of sticking to the limited latent codes, we investigate a set of probability distributions, whose Wasserstein distances from the empirical distribution induced by these latent codes do not exceed a certain radius. The objective function of the deep generator network can therefore be expressed as a minimization problem of the worst-case reconstruction error over a Wasserstein ball of distributions. We further relax the problem for a tractable reformulation with a gradient-based regularizer.
and discuss its connections with GANs. Experimental results on real-world datasets demonstrate the advantages of optimization over the latent Wasserstein ball and the capability of the non-adversarial generator in generating high-quality images as generators from GANs.

2 PRELIMINARIES

We begin with some preliminaries of vanilla GANs [7] and GLO [21]. Then we compare these two generative frameworks and give the motivation of the proposed method.

2.1 Generative Adversarial Networks

GANs [7] provide an effective way to map a simple distribution (e.g., Gaussian) to another complicated one such as natural image distribution. Two networks play a min-max game in the following approach,

\[
\min_{G} \max_{D} \mathbb{E}_{x \sim P_{\text{data}}(x)} \left[ \log(D(x)) \right] + \mathbb{E}_{z \sim P(z)} \left[ \log(1 - D(G(z))) \right],
\]

where \( x \in \mathbb{R}^{w \times h} \) is an image sampled from the real image distribution \( P_{\text{data}}(x) \) and \( z \in \mathbb{R}^{n} \) denotes a noise vector sampled from a known distribution \( P(z) \) (e.g., Gaussian or uniform distribution). In this framework, a discriminator \( D \) distinguishes real samples from generated samples by assigning higher scores to real samples and lower scores to generated samples, respectively. The generator \( G \) is trained to deceive the discriminator \( D \) by generating samples of higher scores. After reaching equilibrium, we expect that the distribution of generated sample \( P_{G} \) could be a nice approximation of the real image distribution \( P_{\text{data}}(x) \).

2.2 Generative Latent Optimization

GANs learned a generator to map a known noise distribution to the real image distribution with the help of the discriminator network. However, given the instability of the training in the min-max game, GANs usually suffer from serious challenges to reach equilibrium. Instead of the adversarial training protocol, GLO [21] proposes to train the generator by providing a non-parametric manner to optimize learnable noise and identify the correspondence between each learned noise vector and the image that it represents. Without the help of discriminator, the generator in GLO can be learned from the following objective function,

\[
\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \left[ \min_{z_i \in \mathcal{Z}} \mathcal{L}(G_{\theta}(z_i), x_i) \right],
\]

where \( x_i \in \mathbb{R}^{w \times h} \) is a real image sampled from the given set of images \( \{x_1, x_2, \ldots, x_N\} \) and \( z_i \in \mathbb{R}^{d} \) is a \( d \)-dimensional latent vector that is paired with the real image \( x_i \). \( \mathcal{L} \) is a loss function to measure the reconstruction error between \( G_{\theta}(z_i) \) and \( x_i \), i.e., the Laplacian pyramid Lap1 loss [22] or the squared-loss. As a result, the training set of GLO can be expressed as \( \{(z_1, x_1), (z_2, x_2), \ldots, (z_N, x_N)\} \). During the training phase of latent vector \( z \), the parameters \( \theta \) are fixed, and the latent vectors are optimized by the gradient derived from the loss function. Thereafter, during the training phase of parameters \( \theta \), the latent vectors are fixed and network parameters \( \theta \) are optimized by gradients. It is well known that the \( d \)-dimensional random vectors sampled from a normal distribution are mostly nearby a sphere, and thus the latent codes \( z \) are projected onto a unit sphere after each iteration during the training. This unit sphere can be considered as the distribution of the latent code.

Given the absence of the discriminator, there is no more adversarial training in GLO. As a result, the training progress of GLO would be much more stable than that of GANs. However, the quality of samples generated by GLO is usually not visually competitive with those generated by GANs. The preliminary reason for this phenomenon is that the generator of GLO mainly focuses on the only \( N \) input data points instead of the distribution of the input. Other data points following or closing to this input distribution have rarely been investigated and would result in samples generated with much lower quality. Experimental results also demonstrate that reconstructed samples tend to have higher fidelity while new latent codes cannot lead to satisfactory generation. In contrast, generator in GANs can access abundant inputs sampled from the entire noise distribution, which are then taken as negative samples and together with the given real samples to learn a discriminator. We are therefore motivated to study how to augment the limited training data in non-adversarial generative networks.

3 LATENT DISTRIBUTION OPTIMIZATION

In this section, we firstly formulate our problem and propose an objective function to optimize the latent distribution. Then we develop a tractable loss function and analyze its effectiveness theoretically. After that, we summarize the proposed optimization pipeline and distinguish it from WGAN-GP.

3.1 Model

Consider \( \{(z_1, x_1), (z_2, x_2), \ldots, (z_N, x_N)\} \) as the training set sampled from \( \hat{P}_N(x) \) of GLO, where input vector \( z_i \) is paired with its corresponding real image \( x_i \). For convenience, we simplify Eq. (2) as following,

\[
\min_{\theta} \mathbb{E}_{z \sim \hat{P}_{N}(z)} \ell_\theta(z),
\]

where \( z_i = \min_{z} \mathcal{L}(G_{\theta}(z), x_i), \hat{P}_N(z) = \hat{P}_N(z|x) \) is the distribution of latent codes \( z \) corresponding to \( x \) in the training set \( \hat{P}_N(x) \) and indicates that \( z_i \) is paired with \( x_i \), and \( \ell_\theta(z) = \mathcal{L}(G_{\theta}(z), x) \). We default \( z \) to be optimized in each round of training. For convenience, in following formulas we mainly focus on the optimization of the parameter \( \theta \) and omit the mark of optimization of \( z \). It is instructive to note that in Eq. (3) the loss function is calculated over the distribution \( \hat{P}_N(z) \) induced by the limited input real images. In contrast, the generator of GANs is trained over noisy vectors freely sampled from a noise distribution \( P(z) \), as shown in Eq. (1). Hence, compared with the established noise distribution \( P(z) \), limited \( \{z, x\} \) pairs in Eq. (3) cannot accurately approximate an ideal distribution \( P(z|x) \), which would degrade the capability of the learned generator. A natural approach to alleviate this problem is to learn the generator over the whole established input distribution \( P(z) \) (e.g., Gaussian distribution), instead of only the training set \( \hat{P}_N(z) \) \( \{(z_1, x_1), (z_2, x_2), \ldots, (z_N, x_N)\} \), which lead us to the following objective function,

\[
\min_{\theta} \mathbb{E}_{z \sim P(z)} \ell_\theta(z),
\]

The main difference between Eq. (3) and Eq. (4) is that Eq. (4) optimizes loss function \( \ell_\theta(z) \) over the training distribution \( P(z) \), which means that the generator \( G_{\theta} \) will produce low reconstruction loss for latent codes sampled from \( P(z) \). For these
latents whose corresponding ground truth image is not available (latent codes outside the training set \( \hat{P}_N(z) \)), lower reconstruction implies higher quality of generation. That is how the generator benefit from Eq. (4). However, as mentioned above, the reconstruction loss over \( P(z) \) cannot be calculated because the absence of ground truth maps for latent codes outside the training set \( \hat{P}_N(z) \), which prevents the objective function Eq. (4) from being tractable. Thus we introduce a distribution set \( \mathcal{B}_e(\hat{P}_N) \) and propose to minimize the loss function over the worst-case distribution \( Q \) within a distribution set \( \mathcal{B}_e(\hat{P}_N) \). \( \mathcal{B}_e(\hat{P}_N) \) is defined as a set consisting of candidate latent distributions that satisfy

\[
\mathcal{B}_e(\hat{P}_N) \triangleq \{ Q \in \mathcal{M}(z) : \mathcal{W}(Q, \hat{P}_N) \leq \epsilon \},
\]

where \( \mathcal{M}(z) \) represents all possible distributions induced by the input \( z \), and \( \epsilon \) should be set to make sure \( \forall Q \in \mathcal{B}_e(\hat{P}_N), \mathcal{W}(Q, \hat{P}_N) \leq \epsilon \) holds. \( \mathcal{W} \) is defined as the Wasserstein distance metric,

\[
\mathcal{W}(Q_1, Q_2) \triangleq \min_{\Pi \in \mathcal{M}(\Pi)} \left\{ \int_{\Pi} \| z_1 - z_2 \| \Pi(\text{d}z_1, \text{d}z_2) \right\},
\]

where \( \Pi \) is a joint distribution of \( z_1 \) and \( z_2 \) whose marginal distributions are \( Q_1 \) and \( Q_2 \), respectively.

The above equation holds if the established distribution \( \hat{P} \) exists in this distribution set \( \mathcal{B}_e(\hat{P}_N) \), and an appropriate value of \( \epsilon \) can satisfy this condition. Then our objective function can be reformulated as,

\[
\min_{\theta} \sup_{Q \in \mathcal{B}_e(\hat{P}_N)} \mathbb{E}_Q[\ell_\theta(z)],
\]

where \( Q \) denotes the latent distribution to be optimized, and \( \hat{P}_N \) represents the training distribution. The Wasserstein distance describes the cost of the optimal mass transportation plan which is decided by \( \Pi \) for moving a mass distribution such as \( Q_1 \) to another \( Q_2 \). In Eq. (6), we measure this cost with the metric \( \| \cdot \| \), which can be an arbitrary norm on \( z_1 \) and \( z_2 \). With the definition of Wasserstein distance metric, the distribution set \( \mathcal{B}_e(\hat{P}_N) \) can be viewed as a Wasserstein ball centred on \( \hat{P}_N \) with a radius determined by \( \epsilon \). Hence the aim of Eq. (8) is to solve the generator \((i.e. \theta)\) by minimizing the maximum loss introduced by a latent distribution \( Q \) that is close to the given distribution \( \hat{P}_N \). The optimization is executed within a Wasserstein ball rather than directly over data points in the training set.

In Eq. (8), we want to have \( \sup_{Q \in \mathcal{B}_e(\hat{P}_N)} \mathbb{E}_Q[\ell_\theta(z)] \) leads to the supremum of \( \mathbb{E}_Q[\ell_\theta(z)] \). However before we illustrate how to solve Eq. (8), we have to first analyze the solvability of this problem. In particular, whether there exists a maximum value (not \(+ \infty\)) of \( \mathbb{E}_Q[\ell_\theta(z)] \) given any \( Q \in \mathcal{B}_e(\hat{P}_N) \). In other words, we expect the loss function to satisfy

\[
\mathbb{E}_Q[\ell_\theta(z)] < +\infty, \quad \forall Q \in \mathcal{B}_e(\hat{P}_N).
\]

Considering the original loss function defined directly over the training distribution \( \hat{P}_N \), we can easily get a trivial conclusion that

\[
\mathbb{E}^{\hat{P}_N}[\ell_\theta(z)] < +\infty.
\]

This is because the training set here is finite and a finite set always has a maximal element. As a result, it allows us to translate Eq. (9) into the following inequality,

\[
\left| \mathbb{E}_Q[\ell_\theta(z)] - \mathbb{E}^{\hat{P}_N}[\ell_\theta(z)] \right| < +\infty, \quad \forall Q \in \mathcal{B}_e(\hat{P}). \tag{11}
\]

Failures to hold the inequality (11) will lead our optimization to be intractable. We employ the following theorem to explain the establishment of Eq. (11), which then suggests the validity of Eq. (9).

**Theorem 1.** Given an \( L \)-Lipschitzian smooth function \( \ell_\theta(z) \), for any distribution \( Q \in \mathcal{B}_e(\hat{P}_N) \), we have,

\[
\left| \mathbb{E}_Q[\ell_\theta(z)] - \mathbb{E}^{\hat{P}_N}[\ell_\theta(z)] \right| < \int Q \parallel \ell_\theta(z_1) - \ell_\theta(z_2) \parallel_\Pi_0(\text{d}z_1, \text{d}z_2) \leq L \cdot \epsilon,
\]

where \( \epsilon \) is the radius of Wasserstein ball, and \( \Pi_0 \) is the joint distribution of \( z_1 \) and \( z_2 \) whose marginals are \( Q \) and \( \hat{P}_N \), respectively.

According to Theorem 1, the left side of Eq. (11) can be upper bounded in terms of the maximum steepness of the function \( \ell_\theta(z) \), which implies that there exists a particular \( Q \) to produce the maximal value of \( \mathbb{E}_Q[\ell_\theta(z)] \). The steepness of the function \( \ell_\theta(z) \) and \( \mathbb{E}_Q[\ell_\theta(z)] \) can be further constrained. The proof of Theorem 1 can be found in Section 6.

Though Theorem 1 indicates that Eq. (8) is solvable, its optimization would be rather challenging. We next proceed to analyze the inner term of Eq. (8), and aim for a tractable reformulation. We first have the following lemma as an intermediate result to bound the supremum of \( \mathbb{E}_Q[\ell_\theta(z)] \).

**Lemma 1.** Consider \( z \in \mathbb{R}^d \) as the input noise vector of generator \( G \), and the distribution \( Q \in \mathcal{B}_e(\hat{P}_N) \) lays in a Wasserstein ball centred at \( \hat{P}_N \) with radius \( \epsilon \). Then for any \( \epsilon \geq 0 \), we have

\[
\sup_{Q \in \mathcal{B}_e(\hat{P}_N)} \mathbb{E}_Q[\ell_\theta(z)] \leq \inf_{\lambda \geq 0} \lambda \epsilon + \frac{1}{N} \sum_{i = 1}^{N} \sup_{Q \in \mathcal{B}_e(\hat{P}_N)} (\ell_\theta(z_i) - \lambda \cdot \| z_i \|),
\]

where \( \{ z_i \}_{i = 1}^{N} \) is the training sample, and \( \lambda \) is a Lagrange multiplier.

Lemma 1 is established under the condition that \( \ell_\theta(z) \) is “proper”, which means that the function \( \ell_\theta(z) \) would not be identically \(+ \infty\) on \( Q \) and has been proved in Theorem 1. Finally, on the basis of Lemma 1, we conduct Theorem 2 to step forward a tractable formulation of the upper bound of the loss function defined in Eq. (8).

**Theorem 2.** Given an \( L \)-Lipschitzian smooth function \( \ell_\theta(z) \), for any distribution \( Q \in \mathcal{B}_e(\hat{P}_N) \), we have,

\[
\sup_{Q \in \mathcal{B}_e(\hat{P}_N)} \mathbb{E}_Q[\ell_\theta(z)] \leq \frac{1}{N} \sum_{i = 1}^{N} (\ell_\theta(z_i)) + \epsilon \cdot L, \quad \forall \epsilon \geq 0,
\]

where \( \hat{z} \) denotes the training sample.

Theorem 2 establishes a connection between the optimization problem over an unknown distribution \( Q \in \mathcal{B}_e(\hat{P}_N) \) and that over the distribution \( \hat{P}_N \) induced by the training sample. Hence, Eq. (8) can be re-formulated as a combination of traditional empirical risk minimization, a new term involving the steepness \( L \) of the
function \( \ell_\theta(z) \), and the radius of a Wasserstein ball. In practice, instead of calculating the steepness of the function \( \ell_\theta(z) \), we turn to constrain the gradient of the function w.r.t. the input noise. Specifically, consider the definition of Lipschitz constant as follow:

\[
\ell_\theta(z) - \ell_\theta(z') \leq \ell \cdot \|z - z'\|.
\] (15)

We choose the smallest value of \( \ell \) that enables the above formula to hold for any \( z \) and \( z' \), which can be expressed as,

\[
\ell := \max_{z} \|\nabla_z \ell_\theta(z)\|.
\] (16)

Eq. (14) can then be rewritten as

\[
\sup_{Q \in \mathcal{B}_r(\mathbb{P}_N)} \mathbb{E}_Q[\ell_\theta(z)] \leq \frac{1}{N} \sum_{i=1}^{N} [\ell_\theta(z_i)] + \epsilon \cdot \max_{z} \|\nabla_z \ell_\theta(z)\|, \forall \epsilon \geq 0.
\] (17)

Eq. (17) shows that the objective function proposed in Eq. (8) can be upper-bounded by the empirical risk and the max gradient term which is also intractable. To address this, we propose to optimize the gradient norm over the training points and obtain our loss function as following,

\[
\mathcal{L}_{GLDO} = \frac{1}{N} \sum_{i=1}^{N} [\ell_\theta(z_i)] + \epsilon \cdot \|\nabla_z \ell_\theta(z_i)\|.
\] (18)

In Eq. (18), we only optimize the gradient norm over the training point instead of over the whole space of latent codes. As a result, there is a gap between the right term of Eq. (17) and Eq. (18). To investigate how much this gap will influence the proposed method, we theoretically analyze the differences between the proposed loss Eq. (18) and the original objective function Eq. (8), and conclude it in Theorem 3 as follows.

**Theorem 3.** Consider \( z \) as the input latent code of generator \( G \), and the distribution \( Q \in \mathcal{B}_r(\mathbb{P}_N) \) lays in a Wasserstein ball centred at \( \mathbb{P}_N \) with radius \( \epsilon \). If \( \ell_\theta \) is Lipschitz continuous or, there are a constant \( \beta \) and a function \( h : \mathcal{Z} \rightarrow \mathbb{R} \) that satisfy

\[
\|\nabla_z \ell_\theta(z_1) - \nabla_z \ell_\theta(z_2)\| \leq h(z_2) \cdot \|z_1 - z_2\|. \quad\text{for any } z \in \mathcal{Z}.
\]

Then let the minimum required radius \( \epsilon_N \geq 0 \) converges to zero as the number of samples \( N \) increases to infinity, we have

\[
\sup_{Q \in \mathcal{H}_r(\mathbb{P})} \mathbb{E}_Q[\ell_\theta(z)] - \mathcal{L}_{GLDO} = o(\epsilon_N),
\] (19)

where \( o(\cdot) \) is the Little-O Notation, and \( \epsilon_N \) depending on the number of samples \( N \) is the minimum radius that satisfies

\[
\mathcal{W}([\mathbb{P}_N, \mathbb{P}_N]) \leq \epsilon_N.
\]

The minimum radius \( \epsilon_N \) tends to decrease as the number of samples \( N \) increases. We can thus expect a smaller \( \epsilon_N \) given a large enough training set. More importantly, Theorem 3 implies that the proposed loss function Eq. (18) is a first-order approximation of the worst-case one. As a result, instead of enforcing the Lipschitz constant which is intractable, enforcing the gradient norm over the training points seems sufficient, and the experimental results also provide evidence for it.

The difficulty in GLO is the limited number of data points for the training. Rather than focusing on the few data points as GLO, we suggest a formulation in terms of the distribution of the latent data points. We aim to optimize over the worst-case distribution within a Wasserstein ball, so that the generator has a better generalization ability for the latent code. According to our theoretical analysis (see Theorem 2), the key is to constrain the minimum Lipschitz constant. Another motivation of the Lipschitz constraint could probably come from the smoothness of the generator. By optimizing the latent codes of GLO (especially initialize the codes with PCA), the resulting latent codes are not well distributed, i.e., there are many holes in the space. This problem could be alleviated by regularizing the GLO to make it smoother. Moreover, there are many ways to realize the Lipschitz constraint. We choose the gradient penalty (GP) for a concrete implementation, because of its proven effectiveness in the field [15], [23]. Other alternatives could include weight decay (WD) and spectral norm (SN) [16]. We further discuss these different regularizers in experiments.

### 3.2 Relationship with Other Generative Models

In this section, we discuss several generative models including both the adversarial manner and the non-adversarial manner.

#### 3.2.1 WGAN-GP

Recently, in addition to implementing Lipschitz continuous in a deep neural network, WGAN-GP [15] proposed a gradient-based penalty on the discriminator network,

\[
\mathcal{L}_{WGAN-GP} = \mathbb{E}_{x \sim \mathbb{P}_x} \left[ (\|\nabla_x D(x)\|_2 - 1)^2 \right],
\] (20)

where \( \mathbb{P}_x \) consists of both real image distribution \( \mathbb{P}_d \) and generated image distribution \( \mathbb{P}_G \). Eq. (20) encourages \( \|\nabla_x D(x)\|_2 \) to go towards 1, and has been proved to successfully constrain the norm of the gradient of discriminator \( |\nabla_x D(x)| \) in experiments.

Though the gradient-penalty is similar to that of WGAN-GP, we emphasize that there are significant differences between our work and WGAN-GP. First of all, we derive this gradient penalty with the aim to optimize the loss function over a Wasserstein ball of distributions, while WGAN-GP obtains the gradient penalty according to an approximated calculation of the Wasserstein distance. Moreover, the proposed method applies this gradient-norm penalty on the generator network rather than the discriminator.
3.2.2 Non-adversarial Methods

GLO only adequately trains over a limited number of latent codes that are most suitable for reconstruction. However, many other latent points have not been sufficiently investigated, and thus the quality of the generated image is not satisfactory. GLANN [24] maps a Gaussian distribution to these well-trained points, thereby avoiding sampling in areas without adequate training.

Similar to GLO, VAE [4] is often plagued by the difficulty of sampling from latent space. There are auto-encoder based generative models following a similar idea as GLANN. For example, GLF [25] proposed an invertible mapping network optimizing the negative log-likelihood to access the latent distribution. 2SVAE [26] proposed to learn the manifold of the training distribution by a VAE in the first stage and then model the latent distribution with a prior Gaussian distribution in the second stage. Compared with GLF and 2SVAE that employ an additional mapping network to sample from the latent space, we solve this problem from another aspect by explicitly focusing on the worst-case distribution of the latent code. Correspondingly, our method will not introduce extra parameters for training. By taking GLO as a reduced autoencoder that has no encoder, we can embed the training strategies of GLF and 2SVAE into GLO and derive two GLO variants as GLO-GLF and GLO-TSV, respectively. In experiments, we compare GLDO with these two methods.

We also consider the contemporaneous work RAE of Ghosh et al. [27], which also introduced a gradient penalty to a generation framework. Our approach also differs from RAE: i) we are motivated to optimize the generator on the worst-case distribution of the latent code in a Wasserstein ball and derive the regularization through a theoretical analysis, while RAE intuitively aims for a regularization to smooth the auto-encoder; ii) RAE is studied within the framework of variational autoencoders (VAEs), but the proposed algorithm aims to tackle the drawbacks of generative latent optimization (GLO); iii) RAE divines into the smoothness of the decoder through its gradient $\nabla D_\theta$, while we start the formulation with the reconstruction loss and consider the gradient of the loss $\nabla \ell_\theta$. These differences thus lead to an in-depth understanding of the algorithm and justify its advantages in experiments. For example, the images produced by VAE-based methods are often less sharp than the proposed GLDO, in particular on large datasets like CelebA. This observation suggests that the prior distribution on the latent space of a VAE may be too strong to fit many images, while vectors in our latent space move freely and use as much space as required to fit the images in the latent space [21]. Our new formulation starting from the perspective of the distribution of the latent codes fixes the drawbacks of GLO and receives further performance improvement.

3.3 Optimization

Our complete algorithm pipeline is summarized in Algorithm 1. Here we explain the training progress and the calculation of the gradient-norm penalty loss in detail. We set the batch size as 1.

Firstly, we assign a latent code $z$ sampled from the established distribution $\mathbb{P}(z)$ (e.g. Gaussian or uniform distribution) to each image $x$; then training set $\mathbb{P}_N$ is built. During the training phase, we sample a training example $\{(x_1, z_1), (x_2, z_2), \ldots, (x_N, z_N)\}$ from the training set $\mathbb{P}_N$. Then we forward propagate the input latent code $z_i$ to the generator $G_\theta$ and obtain the reconstruction loss $\ell_\theta(\hat{z}_i)$ and the gradient norm $\epsilon \cdot ||\nabla z_i, \ell_\theta(\hat{z}_i)||$. Automatic derivation tool integrated in Pytorch is applied here. Now the loss mentioned in Eq. (18) can be easily calculated. Finally, the latent code $z_i$ is updated by the reconstruction loss and the generator parameters $\theta$ are updated by both of them.

GLO provided their experiments with a squared-loss function $\mathcal{L}_2(x, x') = ||x - x'||^2$ and a Laplacian pyramid $\mathcal{L}_p$ loss [22],

$$\mathcal{L}_p(x, x') = \sum_{j=1}^{N} 2^{2j} ||\text{Lap}'(x) - \text{Lap}'(x')||_1,$$  \hspace{1cm} \text{(21)}

where $\text{Lap}'(x)$ indicates the representation of $x$ obtained from the $j$-th level of the Laplacian pyramid [22]. In this paper, we use perceptual loss [28] instead of the Laplacian pyramid loss. Although the perceptual loss outperforms the Laplacian pyramid loss, it suffers from insufficient low-frequency content preservation such as colour, as is the case with the $\mathcal{L}_p$ loss. As a result, we use perceptual loss with a weighted squared-loss.

GLO chooses the input space $\mathcal{Z}$ from the unit sphere rather than a common choice in GANs that is from a Normal distribution on $\mathbb{R}^d$. We observed from the experiments that initialization of the latent vectors with PCA is very crucial for reconstruction, especially on the LSUN [29] dataset. However, initializing noise from PCA tends to produce very poor generation results. Considering the difficulty in getting the noise distribution initialized from the PCA, the noise distribution for the generation in the test stage could be quite different from that in the training period. In contrast, the proposed GLDO can work well by following GANs to choose input noise from a Normal distribution on $\mathbb{R}^d$. We can produce not only reconstruction results of input images but also new generation results that are sufficiently satisfactory under a normal noise distribution initialization.

4 Experiments

In this section, we evaluate the proposed method by conducting comprehensive experiments on five real-world image datasets, MNIST [30], Fashion-MNIST [31], CIFAR-10 [32], CelebA [33] and LSUN [29].

4.1 Experiment Setup

- MNIST [30] consists of a training set with 60,000 examples and a test set with 10,000 examples. Each image is a 28x28 grey-scale image, associated with a label from 10 classes.
- Fashion-MNIST [31] shares the same image size and data structure with MNIST. It contains ten classes of clothing-related grey-scale images such as Pullover, Coat, and Bag.
- CIFAR-10 [32] is an RGB image dataset consisting of 10 categories. It has 60,000 images split into 50,000 training images and 10,000 test images, each image size of 32.
- CelebA [33] consists of 202,599 portraits of celebrities. We first centred crop each image and preprocess it to a size of 64 x 64 pixels. We set the crop size as 160 in most experiments except Sec 4.9.4, where we set it as 108 to obtain a comprehensive evaluation of the proposed method. We follow the method proposed in ProgGAN [17] to generate CelebA at 128 resolution and evaluate the proposed method on the CelebA-128 dataset.
- LSUN [29] contains 10 scene categories, and we conduct our experiments on a bedroom subset. Considering the huge volume of the LSUN-bedroom that contains 3,033,042 images, we conduct experiments with 64-resolution images. We also investigate our method on 128-resolution images where we randomly choose a subset containing 100,000 images as the training set.
the last column of Table 7.

128
the generator of the LSUN dataset and high-resolution images

- \( N \)

scale comparison of GANs \([36]\). The key structure is as follows, i) the network architecture to be the same as the one used in a large-
tion experiments and generation experiments. Specifically, we
DCGAN \([13]\), WGAN-GP \([14]\), and BEGAN \([35]\) in reconstruc-

V AE \([4]\), W AE \([19]\] and GLO \([21]\) and adversarial frameworks
the proposed method with non-adversarial methods PCA \([34]\,

we demonstrate the benchmarks the proposed method performs

Benchmarks and the Network Architecture:
In this section, we evaluate the Inception score with the help of

Fig. 1. Image generation results obtained on MNIST, Fashion-MNIST, CIFAR-10, CelebA (crop size is set as 160), and LSUN datasets.

All the images we used in our experiments are normalized to
colored images in \([0, 1]\). Specifically, we resize the image size of
samples in MNIST and Fashion-MNIST as \(32 \times 32\) so that they

Benchmarks and the Network Architecture: In this section, we
demonstrate the benchmarks the proposed method performs
against and introduce the network structure we used. We compare
the proposed method with non-adversarial methods PCA \([34]\,
VAE \([4]\], WAE \([19]\] and GLO \([21]\) and adversarial frameworks
DCGAN \([13]\), WGAN-GP \([14]\), and BEGAN \([35]\) in reconstruc-
tion experiments and generation experiments. Specifically, we
chose the non-adversarial version (WAE-MMD) in experiments.

For the generator of 32-pixel datasets and CelebA-64, we set
the network architecture to be the same as the one used in a large-
scale comparison of GANs \([36]\). The key structure is as follows, i) nonlinear activation Sigmoid is attached to the end of generators;
ii) the batchsize used in the training process is 64 on all of the
datasets; iii) Adam optimizer with learning rate 0.001 for generator
and 0.003 for input noise vector; iv) noise dimension of 100 for
generator; v) weights initialized from Gaussian: \(N(0; 0.01)\). For
the generator of the LSUN dataset and high-resolution images
with \(128 \times 128\) pixels, we use network architectures reported in
the last column of Table 7.

Evaluation Metrics: We evaluate the proposed method mainly
in terms of three metrics well suited to the image domain.

- Inception score (IS) \([37]\) rewarding high-quality and |
high-variability of samples, can be expressed as:

\[
\exp(\mathbb{E}_x[D_{KL}(p(y|x)||p(y)))], \text{ where } p(y) = \frac{1}{N} \sum_{i=1}^{N} p(y|x^i = G(z^i)) \text{ is the margin distribution and } p(y|x) \text{ is the conditional distribution for samples. In our experiments, we evaluate the Inception score with the help of an pretrained version of Inception model \([38]\) in PyTorch.}

- Frechet Inception Distance (FID) \([39]\) describes the distance
between real and generated distributions and can be computed as follow:

\[
\|\mu_g - \mu_r\|^2 + \text{Tr}(\Sigma_g + \Sigma_r - 2(\Sigma_g\Sigma_r)^{\frac{1}{2}}),
\]

where \((\mu_g, \Sigma_g)\) and \((\mu_r, \Sigma_r)\) indicate the mean and co-variance of embedded samples from generated distribution \(P_g\) and real image distribution \(P_r\), respectively. It is well known that Inception score suffers from a drawback that it is easily fooled by a model which generated only one image per class. In contrast, FID shows more sensitivity to the diversity of samples belonging to the same category. In our experiments, we consider the feature map obtained from the specific layer of the pre-trained inception model as the embedding of the sample. FIDs are calculated with 10,000 randomly chosen train dataset and 10,000 generated samples, and we evaluate the generated results with both IS and FID.

- PRD is an evaluation metric proposed in \([40]\) that measures both

\[
\frac{\alpha(\lambda) = \sum_{\omega \in \Omega} \min(\lambda P(\omega), Q(\omega))},
\]

\[
\beta(\lambda) = \sum_{\omega \in \Omega} \min \left( P(\omega), \frac{Q(\omega)}{\lambda} \right),
\]

Table 1

<table>
<thead>
<tr>
<th>Method</th>
<th>MNIST IS</th>
<th>Fashion-MNIST IS</th>
<th>CIFAR-10 IS</th>
<th>CelebA FID</th>
<th>LSUN-bedroom FID</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLDO (ours)</td>
<td>2.81</td>
<td>3.55</td>
<td>5.87</td>
<td>37.3</td>
<td>42.1</td>
</tr>
<tr>
<td>GLO</td>
<td>1.88</td>
<td>2.18</td>
<td>4.32</td>
<td>56.0</td>
<td>67.5</td>
</tr>
<tr>
<td>VAE</td>
<td>2.35</td>
<td>1.91</td>
<td>3.74</td>
<td>88.2</td>
<td>64.3</td>
</tr>
<tr>
<td>WAE-MMD</td>
<td>2.57</td>
<td>2.73</td>
<td>4.02</td>
<td>57.2</td>
<td>74.6</td>
</tr>
<tr>
<td>DCGAN</td>
<td>3.11</td>
<td>3.88</td>
<td>6.16</td>
<td>33.9</td>
<td>36.2</td>
</tr>
<tr>
<td>WGAN-GP</td>
<td>2.89</td>
<td>3.62</td>
<td>7.46</td>
<td>35.6</td>
<td>35.5</td>
</tr>
<tr>
<td>BEGAN</td>
<td>3.43</td>
<td>3.70</td>
<td>5.62</td>
<td>41.8</td>
<td>47.2</td>
</tr>
</tbody>
</table>

Fig. 1. Image generation results obtained from different models on several datasets.
4.2 Image Generation

GANs are renowned for their strong generation capabilities which enable them to produce very high-quality images. In this section, we compare the proposed method GLDO against not only the non-adversarial models VAE [4], WAE [19] and GLO [21] but also adversarial methods DCGAN [13], WGAN-GP [15], and BEGAN [35]. DCGAN, WGAN-GP, and BEGAN. The GLO and GLDO are trained with simple reconstruction loss, which means that only certain input points can be well trained within the non-adversarial framework. Thus there is no explicit distribution of the inputs in these models. For the generation, we take the same method that GLO does, that fits a single full-covariance Gaussian to the latent codes \( Z \) optimized during the training progress.

We evaluate the proposed method on MNIST [30], Fashion-MNIST [31], CIFAR-10 [32], CelebA [33], and LSUN-bedroom [29] datasets with both Inception score and FID. The quantitative results are presented in Table 1, and the higher Inception score as well as lower FID indicate the higher quality of generated samples. These results show that adversarial methods outperform non-adversarial methods on all datasets. However, we also find that the proposed GLDO outperforms the other non-adversarial models and exhibits competitive performance compared with GAN. On the CelebA dataset, GLDO upgrades FID from 56.0 obtained by GLO to 37.3, which outperforms the 41.8 of BEGAN. Moreover, the FID on the LSUN-bedroom dataset obtained by GLDO is 42.1 and significantly surpasses 47.2 of BEGAN. In Figure 1, we present generated samples on these five datasets obtained by WGAN-GP, WAE, GLO, and GLDO. The quality of generated samples is consistent with the results in Table 1. It shows that samples generated by GLDO enjoy more vibrant colours and sharper edges. By contrast, the samples obtained by GLO are more blurred and even meaningless. For example, the results of GLO on the Fashion-MNIST dataset are more colour and detail than GLO and WAE, especially in the clothes. Figure 2 shows that reconstruction results of GLDO retain more colour and detail than GLO and WAE, especially in the LSUN-bedroom dataset. It proves that the generator is optimized over the worst-case distribution with the help of the proposed method.

### Table 2
Reconstruction results (PSNR) obtained from different models on several datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>PCA</th>
<th>WAE</th>
<th>GAN</th>
<th>GLO</th>
<th>GLDO (ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>21.4</td>
<td>24.5</td>
<td>25.6</td>
<td>26.7</td>
<td><strong>27.1</strong></td>
</tr>
<tr>
<td>Fashion</td>
<td>22.1</td>
<td>30.3</td>
<td>28.4</td>
<td>31.6</td>
<td><strong>31.7</strong></td>
</tr>
<tr>
<td>CelebA</td>
<td>24.6</td>
<td>20.1</td>
<td>22.5</td>
<td><strong>25.2</strong></td>
<td>24.8</td>
</tr>
<tr>
<td>LSUN</td>
<td>23.2</td>
<td>22.7</td>
<td>18.3</td>
<td>22.4</td>
<td><strong>27.8</strong></td>
</tr>
</tbody>
</table>

4.3 Image Reconstruction

We analyze the reconstruction results obtained by PCA, WAE, GAN, GLO, and the proposed method GLDO. To evaluate the reconstruction results, we sample a subset of images from the test set and initialize latent code vectors. Then we implement the back-propagation algorithm to optimize these latent code vectors without fine-tuning the network parameters.

Reconstruction results are reported in Table 2, and reconstruction samples are showed in Figure 2. The results in Table 2 demonstrate that GLDO achieves less reconstruction error than WAE and GLO. Moreover, we also outperform GAN, which shows the advantage of non-adversarial methods that it is easier to find a specific image in generation space. In the Fashion-MNIST dataset, WAE and GLO show similar results with GLDO but fail to reconstruct details of images, such as the texture of clothes. Figure 2 shows that reconstruction results of GLDO retain more colour and detail than GLO and WAE, especially in the LSUN-bedroom dataset. It proves that the generator is optimized over the worst-case distribution with the help of the proposed method.

In this section, we follow the setting of [40] and use the maximum \( F_8 \) and \( F_1 \) scores to summarize the PRD curves, where \( F_i = (1 + t^2)^{i(p/r)^{i/2}} \).

- Peak Signal-to-Noise Ratio (PSNR) is an indicator of the ratio of the maximum possible power of a signal and the destructive noise power that affects the fidelity of its representation, which can be calculated as,

\[
\text{PSNR}(X,Y) = -20 \log_{10} \frac{\max(X)}{\sqrt{\text{MSE}(X,Y)}},
\]

where \( \text{MSE}(X,Y) \) indicates the Mean Squared Error between image \( X \) and its reconstruction result \( Y \), and \( \max(X) \) represents the max value of image \( X \). This evaluation algorithm well suits to evaluate the quality of samples and is close to the results of the subjective quality assessment method.

Fig. 2. Image reconstruction results on FMNIST, CelebA (crop size is set as 160), and LSUN datasets.
4.4 Image Interpolation

Interpolation is a desirable feature of the generative model. Interpolation between a pair of vectors in the latent space is expected to produce semantically meaningful smooth nonlinear interpolation in the image space by the generator. This method of exploration in hidden spaces can effectively illustrate that the generated model successfully fits the distribution of the natural image dataset, rather than just remembering the training samples. We implement the interpolation operation on the CelebA dataset and show a few results in Figures 4.3. At the same time, we quantitatively analyze the performance of GLDO on interpolation experiments employing FID. The results are shown in Table 3.

Figure 4.3 shows that the interpolation results obtained by GLDO present a smooth interpolation process on the CelebA dataset. Moreover, GLDO generates more meaningful interpolation results and higher-quality images than GLO. For example, the first interpolation results of GLDO in Figure 4.3 show a smooth change from a man with glasses to a woman without glasses, and the glasses gradually disappear with the gradual changes in input noises. On the contrary, the results of GLO do not show this process, and even the interpolated results in the middle are difficult to identify. Valid interpolation result means that GLDO has been properly optimized over the latent space, not just empirical distribution. Table 3 also shows that the proposed method dramatically exceeds GLO in FID and is much closed to GANs. As a result, the proposed method provides competitive performance for non-adversarial generation models.

### Table 3

<table>
<thead>
<tr>
<th>Method</th>
<th>Fashion</th>
<th>CelebA</th>
<th>LSUN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCGAN</td>
<td>30.3</td>
<td>41.4</td>
<td>45.1</td>
</tr>
<tr>
<td>WGAN-GP</td>
<td>32.6</td>
<td>44.5</td>
<td>43.0</td>
</tr>
<tr>
<td>GLO</td>
<td>57.4</td>
<td>61.8</td>
<td>69.2</td>
</tr>
<tr>
<td>GLDO (Ours)</td>
<td>38.7</td>
<td>42.7</td>
<td>48.4</td>
</tr>
</tbody>
</table>

4.5 High Resolution Results

Generation of high-resolution images is a challenging task for generative models. We test the proposed method on the CelebA and LSUN datasets with 128 × 128 pixels. We use the latent code dimensionality of 256 for the CelebA dataset, and 512 for the LSUN dataset. Considering the computational cost, we randomly sample 100,000 images from the whole dataset as our training set. We use a learning rate of 0.05 for latent codes $Z$ and 0.005 for parameters of the generator, and learning rate decayed by 0.5 per 10 epochs.

We show high-resolution images generated by GLO and the proposed method in Figure 5. It shows that images generated by the proposed method enjoy more realistic face than GLO. Results obtained on the LSUN dataset also shows that the proposed method generates more meaningful images than GLO. Furthermore, we show four interpolation results in Figure 6, and the smooth change process of the interpolated images can be observed. The success of interpolation in high resolution illustrates that the proposed method does not only remember images existed in the training set but really learns to fit the target image distribution.

4.6 Precision and Recall

The insight of PRD is to characterize precision that how much $P$ can be viewed as a part of $Q$ and recall that how much $Q$ can be viewed as a part of $P$ [40]. PRD contains a two-dimensional score which can reveal in which side the generative model performs badly. We calculate $F_1$ and $F_8$ as the summarization of PRD on the MNIST, FMNIST, CIFAR-10, and CelebA datasets and report them in Table 4, and a higher score indicates better results. The PRD result shows that the proposed method outperforms GLO on both sides, which means that there is more overlap between the target distribution and our generated distribution. This provides evidence that the proposed method is effective to optimize the generator over the latent distribution and generate samples with high quality.

4.7 Convergence Speed

It is well known that the adversarial generation method suffers from problems of unstable training and difficulty in convergence.
Instead, GLO and our approach propose to minimize a definite loss function, which allows our method to converge faster and is hardly affected by convergence difficulties. In this section, we investigated the training process for WGAN-GP, GLO, and GLDO on the Fashion-MNIST dataset. Specifically, we evaluate the generation result of these models with FID score and show the results in Figure 4. The results show that GLO and GLDO training require fewer epochs, while WGAN-GP requires more epochs to converge. In addition, GLDO achieved almost the same FID score as WGAN-GP, far exceeding the GLO generation.

### 4.8 Comparison with mapping network methods

In this part, we compare GLDO with GLANN [24], GLF [25], and 2sVAE [26] on the MNIST and CelebA dataset. As mentioned above, for a fair comparison, we upgrade GLO with the techniques in GLF and 2sVAE and obtain GLO-GLF and GLO-TSV, respectively. Specifically, for GLO-GLF, we add a flow module of GLF to GLO and aim to map the latent code $z$ to a Gaussian distribution. In GLO-TSV, we train a VAE to generate the distribution of the latent code from a Gaussian distribution.

We report the results in Table 5. In terms of the FID score, GLANN obtains 14.2 on the MNIST dataset and 21.5 on the MNIST-Fashion dataset, which are lower than 23.1 and 26.7 of GLDO. On the CelebA dataset, GLDO obtains 36.9 and outperforms the 40.7 of GLANN. Even though GLDO does not introduce an extra network to map a Gaussian distribution to the latent code space like GLANN, GLDO can also obtain competitive generation results, especially on the CelebA dataset.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>GLO</th>
<th>GLO-GLF</th>
<th>GLO-TSV</th>
<th>GLDO</th>
<th>GLANN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>50.1</td>
<td>32.6</td>
<td>42.6</td>
<td>23.1</td>
<td>14.2</td>
</tr>
<tr>
<td>Fashion</td>
<td>52.4</td>
<td>39.8</td>
<td>47.0</td>
<td>26.7</td>
<td>21.5</td>
</tr>
<tr>
<td>CelebA</td>
<td>55.7</td>
<td>45.3</td>
<td>58.4</td>
<td>36.9</td>
<td>40.7</td>
</tr>
</tbody>
</table>

In addition, we find that both GLO-GLF and GLO-TSV obtain performance improvement over GLO on each dataset. For example, on the MNIST dataset, GLO-GLF and GLO-TSV obtain 32.6 and 40.2 scores, while GLO only achieves 50.1. Similar results can be observed on the MNIST-Fashion and CelebA datasets. The performance boost demonstrates that the training strategies of the autoencoder-based model can be successfully transferred to the GLO-based model. Moreover, GLO-GLF obtains 32.6, 39.8, and 45.3 in the MNIST, MNIST-Fashion, and CelebA datasets respectively, which outperform 40.2, 47.0, and 58.4 of GLO-TSV. This is probably because that training a VAE is more difficult than training a Flow network. In addition, GLDO achieves 23.1, 26.7, and 36.9 respectively, that outperform GLO-GLF as well as GLO-TSV on each dataset. Even though GLDO does not have an additional mapping network for help, the proposed can solve the challenges in GLO by investigating the worst-case distribution through the regularization.

### 4.9 Ablation study

In this section, we investigate how the proposed method influenced by the reconstruction loss, the regularization and the hyperparam-
norm of latent variables, we follow [41] to consider
norm of the generators parameters
Lipschitz constant. In particular, for weight decay, we penalize L2
variables (PN), and the aspectual norm (SN) to constrain the
GLO, i.e., weight decay (WD), penalizing the norm of latent
generation are reported in Figure 7.

Figure 7 shows that GLDO_p achieves the best FID and Incep-
tors, respectively.

4.9.1 Reconstruction loss
In the above section, we present generation results obtained by
the proposed model with the help of perceptual loss function and \( L_2 \)
loss on different datasets. However, the GLO model uses the \( \text{Lap}_p \) loss and \( L_2 \) loss for reconstruction on the CelebA dataset. For
a better illustration of the effectiveness of the proposed method,
we investigate the generation result of the GLO model and the
proposed method with the same loss function. For convenience,
we refer to the loss used by GLO as the GLO loss, as is GLDO. We
implement four combinations of model and loss function, which
are GLO models with the \( \text{Lap}_p \) loss (GLO_p), GLO with the
perceptual loss (GLO), GLO with the \( \text{Lap}_p \) loss (GLDO_p)
and GLDO with its original loss (GLDO_p). We investigate these
four models on the CelebA dataset. The FID scores of their
generation are reported in Figure 7.

Figure 7 shows that GLDO_p achieves the best FID and Incep-
tion scores and outperforms GLO_p model with great improvement.
Even with the GLO loss, GLDO model (GLDO) outperforms
GLO model with the perceptual loss (GLO). Generation results
shown in Figure 7 also illustrates it. It is true that the introduction
of perceptual loss improved the performance, but the method of
distribution optimization proposed in this paper provides the main
improvement in the quality of the generated sample.

4.9.2 Choice of Regularization
In this part, we tried three common regularization methods within
GLO, i.e., weight decay (WD), penalizing the norm of latent
variables (PN), and the aspectual norm (SN) to constrain the
Lipschitz constant. In particular, for weight decay, we penalize L2
norm of the generators parameters \( L_{\text{reg}} = \| \theta \|^2_2 \). To penalize
the norm of latent variables, we follow [41] to consider \( L_{\text{reg}} = \| z \|_1 \).
We also attach the spectral norm layer to the convolution layer of
the generator as [16].

Table 6 reports the comparison results on the MNIST and
CelebA datasets. The spectral norm is a powerful and efficient
way to restrict the Lipschitz constant of the generator. Specifically,
GLO-SN obtains 22.6 on the MNIST dataset, which is better than
23.1 of GLDO and 27.6 of GLO-WD. GLO-WD also achieves a
significant improvement over GLO, i.e., 27.6 versus 50.1 and 47.3
versus 55.7 on the MNIST and CelebA datasets, respectively. On
the CelebA dataset, GLDO obtains 36.9, which outperforms 38.2
of GLO-SN and 47.3 of GLO-WD. Overall, spectral norm and
gradient penalty can have comparable performance, and they both
perform better than weight decay. Constraining the Lipschitz con-
stant is indeed helpful in improving the quality of GLO generation.
However, GLO-NP only obtains 55.8 and 73.4 on the MNIST and
CelebA datasets, respectively, and fails to outperform the baseline
method GLO. This could be because that penalizing the norm of
the latent code may dramatically influence the distribution and
representation ability of the latent code.

In addition, inspired by WGAN-GP, incorporating the con-
straints over the convex combinations of pairs is a natural idea
when penalizing the gradient. We denote GLDO-Mix as the model
trained by loss function with gradient penalty applied to convex
combinations of pairs of points. We investigate its performance on
the MNIST and CelebA datasets and obtain 30.4 and 43.5, respec-
tively. These results are much worse than 23.1 and 36.9 of GLDO.
This is because that penalizing the interpolated points have no corresponding
ground-truth data. The reconstruction loss can be estimated only
by referring to those points existing in the training set, which
may lead to huge estimation bias. These results demonstrate the
shortage of the constraints applied between convex combinations
of pairs. As a result, it will not improve results that applying a
gradient penalty to convex combinations of more than two points.

4.9.3 Hyperparameters Analysis
In Eq. (18), a hyperparameter \( \epsilon \) is introduced and influent
the performance of the model. We train our networks with hy-
perparameters of different values and evaluate these networks
generation capabilities. Figure 8 shows the FID scores obtained by
networks with different values of \( \epsilon \) on the CelebA dataset. From
Figure 8 we conclude that a small \( C \) will lead the optimization to
the same situation with that of a big \( k \), and extreme \( C \) value will
TABLE 7
Model architecture for datasets.

<table>
<thead>
<tr>
<th>32×32</th>
<th>CelebA-64</th>
<th>LSUN/128×128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output shape</td>
<td>Layer detail</td>
<td>Output shape</td>
</tr>
<tr>
<td>1024 × 1 × 1</td>
<td>Fully Connected</td>
<td>128 × 1 × 1</td>
</tr>
<tr>
<td>1024 × 1 × 1</td>
<td>BatchNorm1d</td>
<td>1024 × 1 × 1</td>
</tr>
<tr>
<td>1024 × 1 × 1</td>
<td>Leaky ReLU 0.2</td>
<td>1024 × 1 × 1</td>
</tr>
<tr>
<td>128 × 8 × 8</td>
<td>Fully Connected</td>
<td>128 × 16 × 16</td>
</tr>
<tr>
<td>128 × 8 × 8</td>
<td>BatchNorm1d</td>
<td>128 × 16 × 16</td>
</tr>
<tr>
<td>128 × 8 × 8</td>
<td>Leaky ReLU 0.2</td>
<td>128 × 16 × 16</td>
</tr>
<tr>
<td>128 × 8 × 8</td>
<td>Leaky ReLU 0.2</td>
<td>128 × 16 × 16</td>
</tr>
<tr>
<td>64 × 16 × 16</td>
<td>(128, 64, 4, 2, 1)</td>
<td>64 × 32 × 32</td>
</tr>
<tr>
<td>64 × 16 × 16</td>
<td>BatchNorm2d</td>
<td>64 × 32 × 32</td>
</tr>
<tr>
<td>64 × 16 × 16</td>
<td>Leaky ReLU 0.2</td>
<td>64 × 32 × 32</td>
</tr>
<tr>
<td>3 × 32 × 32</td>
<td>(64, 3, 4, 2, 1)</td>
<td>3 × 64 × 64</td>
</tr>
<tr>
<td>3 × 32 × 32</td>
<td>Sigmoid</td>
<td>3 × 64 × 64</td>
</tr>
</tbody>
</table>

TABLE 8
FID scores obtained on the CelebA dataset with 108 crop size.

<table>
<thead>
<tr>
<th>Method</th>
<th>WGAN</th>
<th>WAE</th>
<th>GLO</th>
<th>GLDO (ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generation</td>
<td>36.6</td>
<td>59.7</td>
<td>62.4</td>
<td>38.5</td>
</tr>
<tr>
<td>Reconstruction</td>
<td>–</td>
<td>21.1</td>
<td>26.3</td>
<td>27.2</td>
</tr>
</tbody>
</table>

force the loss function to pay more attention to the gradient-based loss and cannot optimize the reconstruction error. However, the gentle areas at the bottom of curves in Figure 8 indicates that our method enjoys a high tolerance for parameter variation.

4.9.4 Generation with Different Crop Sizes

We further evaluate the proposed method and comparison methods on the CelebA dataset under another crop size, i.e., 108. We report quantitative results in Table 8. We find that with a crop size of 108, most results are better than that of 160 crop size. This may because a smaller crop size will lead to a bigger face in processed images, which are easier for generation and reconstruction tasks. GLDO achieves 36.5 in generation tasks which significantly outperforms WAE of 55.7 and GLO of 62.4, and obtains the best reconstruction result of 27.2. These results demonstrate the consistent superior of GLDO under different crop sizes.

4.10 Architecture

In experiments, we investigate our method in several datasets. There are three kinds of image resolution in these datasets, which are 32, 64, and 128. Three kinds of model structures are utilized for these datasets. Table 7 reports the network structure used in datasets with 32 × 32 resolution images, such as MNIST, Fashion-MNIST, and CIFAR-10, the structure used in the CelebA, whose image resolution are both 64 × 64, and the structure used in the LSUN-bedroom dataset and 128 × 128 resolution image generation task.

In Table 7, the layer detail for convolution layers are displayed in such a format, (input channel, output channel, kernel size, stride, padding). For example, in Table 7 the layer detail of (64, 3, 4, 2, 1) indicates that the input size is 64, output size is 3 (for CIFAR-10) or 1 (for MNIST and Fashion), kernel size is 3 × 3, stride is [1, 1], and padding size is [1, 1, 1, 1].

5 Conclusion

In this paper, we present a non-adversarial generation model that produce competitive results compared to adversarial networks. We analyze the worst-case reconstruction error over a Wasserstein ball of distributions and provide a principled method to optimize a generator over a set of distributions in a Wasserstein ball rather than only the empirical distribution. By introducing a gradient-based penalty to the objective loss function, we prove that the proposed method can successfully provide this desirable optimization. Experiments on real-world image datasets demonstrate that the proposed method outperforms GLO and generates competitive results compared to GANs, especially on CelebA and LSUN datasets, which proves that the proposed model is capable of fitting complex natural image distributions.

6 Proofs

In this section, we provide the proof of Lemma and Theorems proposed in the previous article.

6.1 Proof of Theorem 1

Proof. Following the definition of $\mathbb{E}^Q$ and $\mathbb{E}^P$, we have,

$$
\mathbb{E}^Q[\ell_\theta(z)] = \int_z \ell_\theta(z) Q(dz),
$$

$$
\mathbb{E}^P[\ell_\theta(z)] = \int_z \ell_\theta(z) P(dz).
$$

(25)

Then the left term of Equation (12) can be re-expressed as

$$
\left| \int_z \ell_\theta(z) Q(dz) - \int_z \ell_\theta(z) P(dz) \right| = \left| \int_z \ell_\theta(z) \int_\Pi_0(dz_1, dz_2) - \int_z \ell_\theta(z_2) \int_\Pi_0(dz_1, dz_2) \right|.
$$

(26)
where $\Pi_0$ indicates the joint distribution of $z_1$ and $z_2$ with marginal distributions $Q$ and $P$ respectively. Furthermore, we can upper bound Equation (26) as following,

$$\int_{Z \times Z} |\ell_\theta(z_1) - \ell_\theta(z_2)| \Pi_0(dz_1, dz_2).$$  \hfill (27)$$

Comparing Equation (27) with Wasserstein distance and the constraint of candidate latent distributions, which are defined as

$$\mathcal{W}(Q_1, Q_2), \quad B_x(P) \triangleq \{Q \in \mathcal{M}(Z): \mathcal{W}(Q, P) \leq \epsilon\},$$

respectively. We reformulate Equation (27) as following,

$$\int_{Z \times Z} |\ell_\theta(z_1) - \ell_\theta(z_2)| \cdot \|z_1 - z_2\| \Pi_0(dz_1, dz_2)$$

$$\leq \max_{z \in \mathbb{Q}} \|z_1 - z_2\| \cdot \int_{Z \times Z} \|z_1 - z_2\| \Pi_0(dz_1, dz_2)$$

$$\leq \mathcal{L} \cdot \epsilon.$$

which completes the proof.

\subsection{Proof of Lemma 1}

\textbf{Proof.} With the Definition of the Wasserstein distance which can be expressed as following,

$$\mathcal{W}(Q_1, Q_2) \triangleq \min_{\Pi \in \mathcal{M}(\Pi)} \left\{ \int \|z_1 - z_2\| \Pi(dz_1, dz_2) \right\},$$

we can reformulate the left term in Eq. (12) as

$$\sup_{Q \in B_x(P)} E^Q[\ell_\theta(z)] = \begin{cases} \sup_{\Pi, \mathcal{Q}} \int_{Z} \ell(z) \mathcal{Q}(dz) \\
\text{s.t.} \int_{Z \times Z} \|z_1 - z_2\| \Pi(dz_1, dz_2) \leq \epsilon, \end{cases}$$

where $\Pi$ is a joint distribution of $z_1$ and $z_2$ with marginals $Q$ and $P$, respectively. Following the law of total probability that asserts that we can construct any joint distribution $\Pi$ from the marginal distribution $P \in \mathcal{M}(\Pi)$ and a conditional distribution $Q$, we can further re-express it as follow,

$$\sup_{Q \in B_x(P)} E^Q[\ell_\theta(z)] = \begin{cases} \sup_{\mathcal{Q}, \ell} \frac{1}{N} \sum_{i=1}^{N} \int_{Z} \ell(z) \mathcal{Q}_i(dz) \\
\text{s.t.} \frac{1}{N} \sum_{i=1}^{N} \int_{Z} \|z_1 - z_2\| \mathcal{Q}_i(dz) \leq \epsilon. \end{cases}$$

With the help of standard duality augment, we have

$$\sup_{Q \in B_x(P)} E^Q[\ell_\theta(z)] = \begin{cases} \sup_{Q_i \in \mathbb{Q}} \frac{1}{N} \sum_{i=1}^{N} \int_{Z} \ell_\theta(z) \mathcal{Q}_i(dz) \\
\text{s.t.} \frac{1}{N} \sum_{i=1}^{N} \int_{Z} \|z_1 - z_2\| \mathcal{Q}_i(dz) \leq \epsilon. \end{cases}$$

which completes the proof.

\subsection{Proof of Theorem 2}

\textbf{Proof.} Following the result of Lemma 1, we re-express the left term of Theorem 2 as

$$\sup_{Q \in B_x(P)} E^Q[\ell_\theta(z)] \leq \inf_{\lambda \geq 0} \lambda \epsilon + \frac{1}{N} \sum_{i=1}^{N} \sup_{z \in Z} (\ell_\theta(z) - \lambda \cdot \|z - z_i\|)$$

$$\leq \inf_{\lambda \geq 0} \lambda \epsilon + \frac{1}{N} \sum_{i=1}^{N} \sup_{z \in Z} (\ell_\theta(z) - \ell_\theta(z_i) + (\mathcal{L} - \lambda) \cdot \|z - z_i\|).$$

(34)

Considering the $\ell_\theta(z) - \ell_\theta(z_i) \leq \mathcal{L} \cdot \|z - z_i\|$ holds, we have

$$\sup_{Q \in B_x(P)} E^Q[\ell_\theta(z)] \leq \inf_{\lambda \geq 0} \lambda \epsilon + \frac{1}{N} \sum_{i=1}^{N} \ell_\theta(z_i) + (\mathcal{L} - \lambda) \cdot \epsilon$$

$$= \frac{1}{N} \sum_{i=1}^{N} (\ell_\theta(z_i)) + \mathcal{L} \cdot \epsilon.$$

The proof completes.

\subsection{Proof of Theorem 3}

In this part, we firstly proof the right part of this theorem which is an upper bound of $E^Q[\ell(z)] - E^{\hat{P}_N}[\ell(z)]$. Then we provide the proof of a lower bound of it. By combining them, the proof of theorem 2 is completed.

\textbf{Proof.} In Lemma 1 we show that,

$$\sup_{Q \in B_x(P)} E^Q[\ell_\theta(z)] \leq \inf_{\lambda \geq 0} \lambda \epsilon + \frac{1}{N} \sum_{i=1}^{N} \sup_{z \in Z} (\ell_\theta(z) - \ell_\theta(z_i)) - \lambda \cdot \|z - z_i\|_{\alpha}.$$

(36)

As the metric $\|\cdot\|$ can be any norm, such as $\|\cdot\|_{\alpha}$, the Wasserstein is easy to be reformulated as follows:

$$\mathcal{W}(Q_1, Q_2) = \min_{\Pi \in \mathcal{M}(\Pi)} \left\{ \int \|z_1 - z_2\| \Pi(dz_1, dz_2) \right\}.$$  \hfill (37)

Plugging Eq. (37) into Eq. (36) gives us

$$\sup_{Q \in B_x(P)} E^Q[\ell_\theta(z)] - E^{\hat{P}_N}[\ell_\theta(z)] \leq$$

$$\inf_{\lambda \geq 0} \lambda \epsilon + \frac{1}{N} \sum_{i=1}^{N} \sup_{z \in \mathbb{Q}} (\ell_\theta(z) - \ell_\theta(z_i)) - \lambda \cdot \|z - z_i\|_{\alpha},$$

where $z \sim Q$ and $z' \sim \hat{P}_N$, then we consider a upper bound that

$$\sup_{a \in \mathbb{A}} (\ell(z) - \ell(z_i)) \leq \sup_{a \in \mathbb{A}} (\|\nabla_a \ell(z)\|_{\alpha} \cdot \|z - z_i\|_1 + h(z_i) \cdot \|z - z_i\|_{\beta+1}$$

$$- \lambda \cdot \|z - z_i\|_{\alpha})$$

$$\leq \sup_{a \in \mathbb{A}} (\|\nabla_a \ell(z)\|_{\alpha} \cdot \|z - z_i\|_1 + h(z_i) \cdot \|z - z_i\|_{\beta+1}$$

$$- \lambda \cdot \|z - z_i\|_{\alpha} + C \cdot \|z - z_i\|_{\gamma+1})$$

$$\leq \sup_{a \in \mathbb{A}} (\|\nabla_a \ell(z_i)\|_{\alpha} \cdot \xi + h(z_i) \cdot \xi_{\beta+1} + C \cdot \xi_{\gamma+1} - \lambda \cdot \xi^\alpha),$$

(39)
where $0 \leq C$, $1 < \gamma < \beta$ and $\xi := \|z - z_i\|$. Following Young’s inequality for products that $ab \leq \frac{a^\alpha}{\beta} + \frac{b^\beta}{\alpha}$ we set $p = \frac{\alpha-1}{\alpha-\beta}$, $q = \frac{\alpha-1}{\beta}$ satisfying $\frac{1}{p} + \frac{1}{1} = 1$ and $a = \varphi^\alpha \xi^{1/p}$, $b = \varphi^{-\beta/q} \xi^{1/q} / \varphi$. Then for any $t > 0$ and $\varphi > 0$, it holds that

$$\xi^\alpha + \|z - z_i\| \leq \frac{\alpha-1}{\alpha-\beta} \varphi \xi + \frac{\alpha-1}{\alpha-\beta} \varphi - \frac{\alpha-1}{\alpha-\beta} \xi. $$

(40)

Replacing $\xi^\alpha + \|z - z_i\|$ with the last term of Eq. (40), it gives us

$$\sup_{i \geq 0} \left\{ \|\nabla z_i(z_i)\| + \xi + h(z_i) + \xi + C \cdot \xi^\alpha - \lambda \cdot \xi^\alpha \right\}$$

$$\leq \sup_{i \geq 0} \left\{ \|\nabla z_i(z_i)\| + \frac{\alpha-1}{\alpha-\beta} \frac{\alpha-1}{\alpha-\beta} h(z_i) + \frac{\alpha-1}{\alpha-\beta} C \cdot \varphi^\beta \cdot \xi^\alpha \right\}$$

$$- \frac{\alpha-1}{\alpha-\beta} \left( \frac{\alpha-1}{\alpha-\beta} \right) \frac{\alpha-1}{\alpha-\beta} - \frac{\alpha-1}{\alpha-\beta} C \varphi^\beta \cdot \xi^\alpha$$

$$\leq \sup_{i \geq 0} \left\{ \mathcal{G}_\varphi(z_i) \cdot \xi + (\lambda - \mathcal{N}_\varphi) \cdot \xi^\alpha \right\},$$

(41)

where $\mathcal{G}_\varphi(z_i) = \|\nabla z_i(z_i)\| + \frac{\alpha-1}{\alpha-\beta} h(z_i) + \frac{\alpha-1}{\alpha-\beta} C \cdot \varphi^\beta$ and $\mathcal{N}_\varphi = \lambda - \frac{\alpha-1}{\alpha-\beta} \cdot h(z_i) \cdot \varphi^\beta - \frac{\alpha-1}{\alpha-\beta} C \cdot \varphi^\beta$. Considering the value of Eq. (41) is $\infty$ when $\lambda \leq \mathcal{N}_\varphi$, we solve Eq. (41) over $\xi$ and conclude that

$$\mathcal{E}^\mathcal{Q}_\mathcal{F}(f(z)) - \mathcal{E}^{\mathcal{F}_\mathcal{N}}(f(z))$$

$$\leq \inf_{\lambda \geq \mathcal{N}_\varphi} \left\{ \lambda \alpha^\alpha + \frac{\alpha^{\alpha-1}}{\alpha-\beta} (\alpha-1) (\lambda - \mathcal{N}_\varphi) - \frac{\alpha^{\alpha-1}}{\alpha-\beta} - \frac{\alpha^{\alpha-1}}{\alpha-\beta} \frac{\alpha}{\alpha-\beta} \right\}$$

$$\leq \epsilon \|
abla z_i(z_i)\|^{\alpha^\alpha} + \mathcal{N}_\varphi \epsilon^{\alpha}.$$

(42)

Plugging $\mathcal{G}_\varphi$ and $\mathcal{N}_\varphi$ into Eq. (42) and solving the minimization problem on $\mathcal{F}$, we obtain the upper bound of $\sup_{Q \in \mathcal{B}_\mathcal{F}^{\mathcal{F}_\mathcal{N}}(\mathcal{F})} \mathcal{E}^\mathcal{Q}_\mathcal{F}(f(z)) - \mathcal{E}^{\mathcal{F}_\mathcal{N}}(f(z))$ as follows:

$$\sup_{Q \in \mathcal{B}_\mathcal{F}^{\mathcal{F}_\mathcal{N}}(\mathcal{F})} \mathcal{E}^\mathcal{Q}_\mathcal{F}(f(z)) - \mathcal{E}^{\mathcal{F}_\mathcal{N}}(f(z))$$

$$\leq \epsilon \|
abla z_i(z_i)\|^{\alpha^\alpha} + \mathcal{N}_\varphi \epsilon^{\alpha}.$$

(43)

Next we step to the left part of Theorem 2. We firstly bring up a lower bound of $\sup_{Q \in \mathcal{B}_\mathcal{F}^{\mathcal{F}_\mathcal{N}}(\mathcal{F})} \mathcal{E}^\mathcal{Q}_\mathcal{F}(f(z)) - \mathcal{E}^{\mathcal{F}_\mathcal{N}}(f(z))$ as follows:

$$\sup_{z_i \in \mathbb{Z}} \left\{ \frac{1}{N} \sum_{i=1}^{N} \|\nabla z_i(z_i)\| \cdot \|z_i - z_i'\|^{\alpha^\alpha} \leq \epsilon \right\}$$

$$\geq \sup_{z_i \in \mathbb{Z}} \left\{ \frac{1}{N} \sum_{i=1}^{N} \|\nabla z_i(z_i)\| \cdot \|z_i - z_i'\| - h(z_i) \cdot \|z_i - z_i'\|^{\beta+1} \right\}$$

$$: \left( \frac{1}{N} \sum_{i=1}^{N} \|z_i - z_i'\|^{\beta+1} \right) \leq \epsilon$$

$$\geq \sup_{z_i \in \mathbb{Z}} \left\{ \frac{1}{N} \sum_{i=1}^{N} \|\nabla z_i(z_i)\| \cdot \|z_i - z_i'\|$$

$$: \left( \frac{1}{N} \sum_{i=1}^{N} \|z_i - z_i'\|^{\beta+1} \right) \leq \epsilon$$

$$- \sup_{z_i \in \mathbb{Z}} \left\{ \frac{1}{N} \sum_{i=1}^{N} h(z_i) \cdot \|z_i - z_i'\|^{\beta+1} \right\}$$

$$: \left( \frac{1}{N} \sum_{i=1}^{N} \|z_i - z_i'\|^{\beta+1} \right) \leq \epsilon.$$