

# Improving Generative Adversarial Networks with Local Coordinate Coding

Jiezhang Cao\*, Yong Guo\*, Qingyao Wu, Chunhua Shen, Junzhou Huang, Mingkui Tan†

**Abstract**—Generative adversarial networks (GANs) have shown remarkable success in generating realistic data from some predefined prior distributions (*e.g.*, Gaussian noises). However, such prior distributions are often independent of real data and thus may lose semantic information (*e.g.*, geometric structure or content in images) of data. In practice, the semantic information might be represented by some latent distribution learned from data. However, such latent distribution may incur difficulties in data sampling for GAN methods. In this paper, rather than sampling from the predefined prior distribution, we propose a local coordinate coding GAN, termed LCCGAN-v1, to improve the performance of image generation. First, we propose a local coordinate coding (LCC)-based sampling method in LCCGAN-v1 to sample meaningful points from the latent manifold. With the LCC sampling method, we can explicitly exploit the local information on the latent manifold and thus produce new data with promising quality. Second, we propose an improved version, namely LCCGAN-v2, by introducing a higher-order term in the generator approximation. This term is able to achieve better approximation and thus further improve the performance. More critically, we derive the generalization bound for both LCCGAN-v1 and LCCGAN-v2 and prove that a small-dimensional input is sufficient to achieve good generalization performance. Extensive experiments on four benchmark datasets demonstrate the superiority of the proposed method over existing GAN methods.

**Index Terms**—local coordinate coding (LCC), generative adversarial networks (GANs), latent distribution, generalization bound

## 1 INTRODUCTION

GENERATIVE adversarial networks (GANs) [1] have been successfully applied in many computer vision tasks, such as image generation [2], [3], [4], [5], [6], [7], [8], video prediction [9], [10], image translation [11], [12] and domain adaptation [13], [14], [15], [16]. In general, a GAN consists of a generator and a discriminator to play a two-player game. Specifically, the generator learns from a simple prior distribution (*e.g.*, Gaussian distribution [1]) to produce plausible samples to fool the discriminator, while the discriminator distinguishes the fake samples from the real data. Recently, many studies [2], [3], [17], [18], [19] have been proposed to improve the performance of GANs, which, however, suffer from three limitations.

First, many GAN methods employ some simple prior distribution, such as Gaussian distributions [1] and uniform distributions [17]. However, such predefined prior distributions are often independent of the data distribution, and these methods may produce images with distorted structures without sufficient semantic information. Although such semantic information can be represented by some latent distributions, *e.g.*, extracting embeddings using an autoencoder [20], how to conduct sampling from these distributions still remains largely unsolved in GANs.

Second, the correspondence between the semantic information and the dimension of latent distribution is not yet fully exploited. Most GAN methods [1], [3] use a global coordinate system to represent the data manifold and employ random noises as the

codings to generate data (See Fig. 1). However, these methods fail to study the underlying geometry and capture the local information of data. As a result, it is possible to sample meaningless points in such global coordinate systems. For this issue, determining how to exploit the semantic information of data and such correspondence is a very challenging problem.

Third, the generalization ability of GANs *w.r.t.* the dimension of the latent distribution remains unclear. In practice, the performance of GANs is often sensitive to the dimension of the latent distribution [2]. Unfortunately, it is hard to define the generalization of GANs and analyze the dimensionality of the latent distribution, since the prior distribution is independent of real data. Therefore, study on the role of the dimension of the latent distribution and investigation of its impact on the generalization ability become increasingly important.

In this paper, relying on the manifold assumption on images [21], [22], we propose a novel generative model using local coordinate coding (LCC) [23] to improve the performance of GANs. Specifically, we first employ an autoencoder to learn the embeddings lying on the latent manifold to capture the semantic information in real data. Then, we develop a new LCC sampling method for training GANs by exploiting the local information on the latent manifold. For convenience, we term this method LCCGAN-v1, which appeared in [2].

Based on LCCGAN-v1, we propose an improved version, namely LCCGAN-v2, by introducing a higher-order term to further improve the approximation of generative models. LCCGAN-v2 shows more stable training behavior and obtains better performance than LCCGAN-v1. More critically, we analyze the generalization performance for both LCCGAN-v1 and LCCGAN-v2 and prove that a low-dimensional input is sufficient to achieve good generalization performance.

The contributions of this paper are summarized as follows.

Jiezhang Cao, Yong Guo, Qingyao Wu, and Mingkui Tan are with Pazhou Laboratory; School of Software Engineering, South China University of Technology. E-mail: {seccaojiezhang, guo.yong}@mail.scut.edu.cn {qyw, mingkui-tan}@scut.edu.cn

Chunhua Shen is with The University of Adelaide, Australia. E-mail: chunhua.shen@adelaide.edu.au

Junzhou Huang is with Tencent AI Lab, China; University of Texas at Arlington, America. E-mail: joehuang@tencent.com

\* Authors contributed equally.

† Corresponding author.

- We propose an LCC sampling method for GANs to capture the local information of real data. With the LCC sampling method, the proposed scheme, namely LCCGAN-v1, is able to sample meaningful points from the latent manifold to generate new data.
- Based on LCCGAN-v1, we propose an improved version LCCGAN-v2 by introducing a higher-order term to further improve the approximation of generative models. LCCGAN-v2 shows more stable training behavior and better performance than our preliminary work LCCGAN-v1.
- We derive the generalization bound for both LCCGAN-v1 and LCCGAN-v2 based on Rademacher complexity of the discriminator set and the error *w.r.t.* the intrinsic dimensionality of the manifold. In particular, we theoretically prove that a low-dimensional input is sufficient to achieve good generalization performance. Extensive experiments on several real-world datasets demonstrate the superiority of the proposed method over several baseline methods.

## 2 RELATED WORK

**Generative adversarial networks.** Recently, generative adversarial networks (GANs) [1] have been successfully applied to many computer vision tasks, such as image generation [2], [3], [4], video prediction [9], [10], image translation [11], [12] and domain adaptation [13], [14], [15], [16]. Most GAN methods (*e.g.*, DCGAN [1], WGAN-GP [24] and Progressive GAN [18]) employ global coordinate systems with some prior distribution (such as Gaussian distributions or uniform distributions) to generate samples. Unfortunately, using such global coordinate systems may fail to learn the underlying geometry of data and thus often samples meaningless points to generate distorted data. Moreover, such prior distributions are independent of the data distributions, which may lose semantic information of real data and lead to difficulties in analyzing the dimension of latent space. To address this, LGAN [25] uses local coordinate systems and presents a local generator whose input is sampled from a mixture of Gaussian noises with the discrete distribution. As a result, LGAN is able to generate images of good quality. However, this method is difficult to explore the correlation between the semantic information of real data and the dimension of a latent distribution. Recently, LCCGAN-v1 [2] has employed a local coordinate system to exploit such correlation and improve the performance of GANs.

Furthermore, some generative models conduct sampling via some learned posterior distribution. For example, the variational autoencoder (VAE) [26] combines a generative model and an approximate inference model to perform posterior inference. Moreover, the Wasserstein autoencoder (WAE) [27] builds a real data distribution by minimizing a term of the Wasserstein distance between the model distribution and the target distribution, encouraging the encoded training distribution to match the prior. In addition, the adversarial autoencoder (AAE) [28] matches the aggregated posterior distribution to the prior distribution to perform variational inference. However, these methods cannot directly conduct sampling on the posterior distribution. Moreover, since they globally parameterize the manifold, they would lose local semantic information or have difficulty accessing the local geometry along the manifold.

**Generalization analysis of GAN methods.** Recently, several generalization analysis methods are proposed to understand and

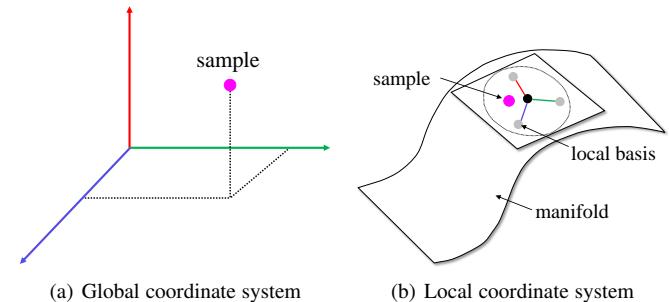


Fig. 1. Comparisons of the global and local coordinate system. (a) In the global coordinate system, most GAN methods are hard to learn the underlying geometry of real data. Therefore, they often sample meaningless points in such a global coordinate system. (b) In the local coordinate system, GAN methods are able to learn the underlying geometry and capture the local information of real data. As a result, they can sample a new point with the semantic information.

improve the generalization performance of GAN methods. For example, Dziugaite *et al.* [29] propose adversarial learning using maximum mean discrepancy (MMD) and provide generalization bounds of MMD nets related to the fat-shattering dimension of the class of generators. Moreover, Thanh-Tung *et al.* [30] show that discriminators trained on discrete datasets with the original GAN loss would fail to guarantee good generalization performance of GANs and thus provide a zero-centered gradient penalty to improve the generalization of the discriminator. In addition, Jiang *et al.* [31] derive a generalization bound under spectrum control based on the PAC-learning framework and prove that the spectrum control is able to improve the generalization ability of GAN models. However, these generalization analysis methods do not understand the generalization performance of GAN methods well from the rigorous mathematical definition.

To address this shortcoming, Arora *et al.* [32] formally provide a definition of the generalization for GAN methods, and prove that the neural net distance is able to guarantee the generalization performance. However, Jensen-Shannon divergence and Wasserstein distance do not generalize with any polynomial number of examples because the expected distance is not reflected by the empirical distance. Based on the definition of generalization, Zhang *et al.* [33] use different evaluation metrics to develop generalization bounds between the true distribution and learned distribution, and prove that the set of discriminators should be large enough to identify the true distribution and small enough surpass memorizing samples. Furthermore, Cao *et al.* [2] employ the neural net distance to define the generalization *w.r.t.* the dimension of the latent distribution. In addition, they develop a generalization bound related to the Rademacher complexity of the set of the discriminator, and prove that a small-dimensional input is sufficient to achieve good generalization. They prove that a small-dimensional input is sufficient to achieve good generalization performance. Recently, Cao *et al.* [34] extend the definition of generalization of GANs to the case of multiple domains. However, this analysis method is hard to understand the generalization performance of GANs *w.r.t.* the dimension of the latent distribution. To better understand the generalization performance of GAN methods, we further study the relationship between the generalization and the dimension of the latent distribution in this paper.

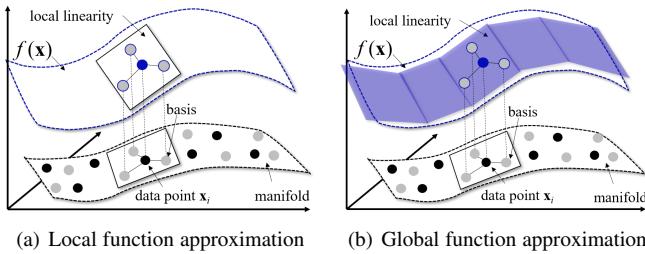


Fig. 2. A geometric view of local coordinate coding. Given a set of local bases, if data lie on a manifold, a nonlinear function  $f(\mathbf{x})$  can be locally approximated by a linear function w.r.t. the coding. Given all bases,  $f(\mathbf{x})$  can be globally approximated.

### 3 PRELIMINARIES

**Notation.** Throughout the paper, we use bold lower-case letters (*e.g.*,  $\mathbf{x}$ ) to denote vectors and bold upper-case letters (*e.g.*,  $\mathbf{X}$ ) to denote matrices. We denote by the superscript  $\top$  the transpose of a vector or matrix, and denote by  $\|\cdot\|$  the Euclidean norm ( $\ell_2$  norm) on  $\mathbb{R}^d$ , *i.e.*,  $\|\mathbf{x}\| = \|\mathbf{x}\|_2 = (\sum_i x_i^2)^{1/2}$ .

#### 3.1 Local Coordinate Coding

We first introduce the following definitions of Lipschitz smoothness and local coordinate coding (See Fig. 2), and then use them to develop our proposed GAN method.

**Definition 1 (Lipschitz Smoothness [35])** A function  $f(\mathbf{x})$  in  $\mathbb{R}^d$  is  $(L_{\mathbf{x}}, L_f, L_{\nu})$ -Lipschitz smooth if

- 1)  $\|\nabla f(\mathbf{x})^\top(\mathbf{x}' - \mathbf{x})\| \leq L_{\mathbf{x}} \|\mathbf{x} - \mathbf{x}'\|$ ,
  - 2)  $\|f(\mathbf{x}') - f(\mathbf{x}) - \nabla f(\mathbf{x})^\top(\mathbf{x}' - \mathbf{x})\| \leq L_f \|\mathbf{x} - \mathbf{x}'\|^2$ ,
  - 3)  $\|f(\mathbf{x}') - f(\mathbf{x}) - \frac{1}{2}(\nabla f(\mathbf{x}') + \nabla f(\mathbf{x}))^\top(\mathbf{x}' - \mathbf{x})\| \leq L_{\nu} \|\mathbf{x} - \mathbf{x}'\|^3$ ,
- where  $L_{\mathbf{x}}, L_f, L_{\nu} > 0$ .

In Definition 1, the Lipschitz constants  $L_{\mathbf{x}}$ ,  $L_f$  and  $L_{\nu}$  are finite if  $f(\mathbf{x})$ , the derivative  $\nabla f(\mathbf{x})$  and the Hessian of  $f(\mathbf{x})$  are Lipschitz, respectively. These three Lipschitz constants measure different levels of smoothness of  $f(\mathbf{x})$ , *i.e.*, when  $\|\mathbf{x} - \mathbf{x}'\|$  is small,  $L_{\mathbf{x}}$  measures how well  $f(\mathbf{x})$  can be approximated by a constant function.  $L_f$  measures how well  $f(\mathbf{x})$  can be approximated by a linear function in  $\mathbf{x}$ , and  $L_{\nu}$  measure how well  $f(\mathbf{x})$  can be approximated by a quadratic function in  $\mathbf{x}$ .

**Definition 2 (Coordinate Coding [23])** Given a coordinate coding  $(\gamma, \mathcal{C})$ , where  $\mathcal{C} \subset \mathbb{R}^d$  is a set of anchor points (bases), and let  $\gamma$  be a map of  $\mathbf{x} \in \mathbb{R}^d$  to  $[\gamma_v(\mathbf{x})]_{v \in \mathcal{C}} \in \mathbb{R}^{|\mathcal{C}|}$  such that  $\sum_v \gamma_v(\mathbf{x}) = 1$ , then the physical approximation of  $\mathbf{x} \in \mathbb{R}^d$  is

$$\mathbf{r}(\mathbf{x}) = \sum_{v \in \mathcal{C}} \gamma_v(\mathbf{x}) \mathbf{v}.$$

In Definition 2, any point  $\mathbf{x}$  can be approximated by a linear combination of a set of anchor points  $\mathbf{v} \in \mathcal{C}$ . This definition is important for local coordinate coding.

#### 3.2 Latent Manifold

Based on the manifold assumption, high-dimensional data (*e.g.*, images) in the real world often lie on some low dimensional manifold [21], [22]. Formally, the latent manifold and its intrinsic dimensionality can be defined as follows.

**Definition 3 (Latent Manifold [23])** A subset  $\mathcal{M}$  embedded in the latent space  $\mathbb{R}^{d_B}$  is called a smooth manifold with an **intrinsic dimension**  $d := d_{\mathcal{M}}$ , if there exists a constant  $c_{\mathcal{M}}$ , such that given any  $\mathbf{h} \in \mathcal{M}$ , there are  $d$  bases (tangent directions)  $\mathbf{v}_1(\mathbf{h}), \dots, \mathbf{v}_d(\mathbf{h}) \in \mathbb{R}^{d_B}$  so that  $\forall \mathbf{h}' \in \mathcal{M}$ :

$$\inf_{\gamma \in \mathbb{R}^d} \left\| \mathbf{h}' - \mathbf{h} - \sum_{j=1}^d \gamma_j \mathbf{v}_j(\mathbf{h}) \right\| \leq c_{\mathcal{M}} \|\mathbf{h}' - \mathbf{h}\|^2, \quad (1)$$

where  $\gamma = [\gamma_1, \dots, \gamma_d]^\top$  is the local coding of a latent point  $\mathbf{h}$  using the corresponding bases.

Based on Definition 3, we seek to learn a latent manifold  $\mathcal{M}$  embedded in the latent space  $\mathbb{R}^{d_B}$  to build a relationship between the latent distribution and the data distribution. To this end, one simple approach is to use some manifold learning method, such as an autoencoder [20], to capture the semantic information of real data. Specifically, given  $N$  training data  $\{\mathbf{x}_i\}_{i=1}^N$ , we can use an Encoder( $\cdot$ ) to extract their corresponding embeddings  $\{\mathbf{h}_i\}_{i=1}^N$ , where  $\mathbf{h}_i = \text{Encoder}(\mathbf{x}_i)$ ,  $i \in 1, \dots, N$ . Thus, we are able to model the latent distribution relying on real data.

#### 3.3 Generative Adversarial Networks

In existing studies [1], [3], the Jensen-Shannon divergence and Wasserstein distance are often used to measure the similarity between two different distributions. However, these measures cannot generalize with any polynomial number of examples [32]. To guarantee the generalization performance of GANs, we apply the following neural network distance [32] to measure the divergence between two distributions.

**Definition 4 (Neural Network Distance [32])** Let  $\mathcal{F}$  be a set of neural networks from  $\mathbb{R}^d$  to  $[0, 1]$  and  $\phi$  be a concave measure function; then, for  $D \in \mathcal{F}$ , the neural network distance w.r.t.  $\phi$  between two distributions  $\mu$  and  $\nu$  can be defined as

$$d_{\mathcal{F}, \phi}(\mu, \nu) = \sup_{D \in \mathcal{F}} \left| \mathbb{E}_{\mathbf{x} \sim \mu} [\phi(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{x} \sim \nu} [\phi(1 - D(\mathbf{x}))] \right| - \phi_c,$$

where  $\phi_c = 2\phi(1/2)$  is a constant with the given function  $\phi(\cdot)$ . For simplicity, we omit the constant  $\phi_c$ .

**Objective function of general GANs.** Given a Generator  $G_u$  and a Discriminator  $D_v$  parameterized by  $u \in \mathcal{U}$  and  $v \in \mathcal{V}$ , respectively, where  $\mathcal{U}$  and  $\mathcal{V}$  are parameter spaces, and using the definition of the neural network distance, the objective function of GANs can be defined as:

$$\min_{u \in \mathcal{U}} \max_{v \in \mathcal{V}} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_{\text{real}}} [\phi(D_v(\mathbf{x}))] + \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_{G_u}} [\phi(1 - D_v(\mathbf{x}))],$$

where  $\phi : [0, 1] \rightarrow \mathbb{R}$  is any monotone function,  $\mathcal{D}_{\text{real}}$  is the real distribution and  $\mathcal{D}_{G_u}$  is the distribution generated by  $G_u$ .

For example, when  $\phi(t) = \log(t)$  and  $\mathcal{F} = \{f: \mathbf{x} \rightarrow [0, 1]\}$ , then minimizing  $d_{\mathcal{F}, \phi}(\mu, \nu)$  is equivalent to the original GAN objective. When  $\phi(t) = t$ ,  $f \in \mathcal{F}$  and  $f$  is 1-Lipschitz, then  $d_{\mathcal{F}, \phi}(\mu, \nu)$  corresponds to the Wasserstein distance.

By optimizing the objective function, the generator tries to produce realistic data to fool the discriminator, and the discriminator tries to distinguish between the generated data and real data.

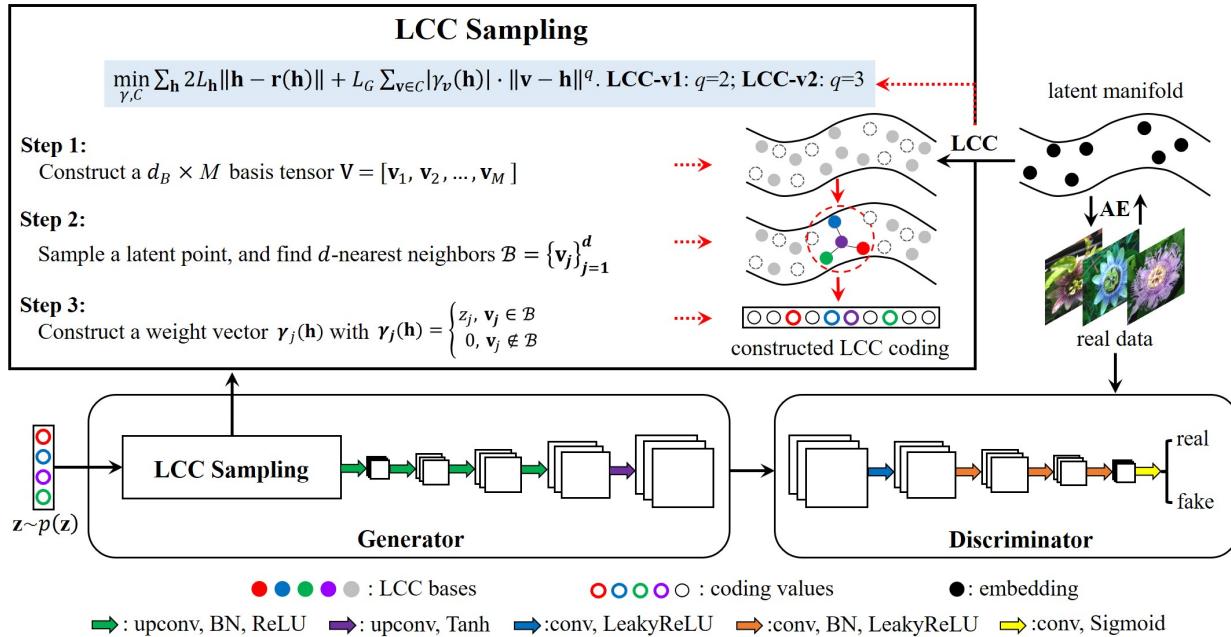


Fig. 3. The scheme of the proposed LCCGAN. We use an autoencoder to learn the embeddings on the latent manifold from real data. We minimize the objective function of LCC with different  $q$  to learn a set of bases such that the LCC sampling can be conducted. Specifically, we train LCCGAN-v1 with  $q = 2$  and LCCGAN-v2 with  $q = 3$ . Thus, LCCGAN takes as input the constructed LCC codings to generate new data.

## 4 ADVERSARIAL LEARNING WITH LCC

In this section, we seek to improve GANs by exploiting the local coordinate coding (LCC). The overall structure of the proposed method, called LCCGAN, is illustrated in Fig. 3.

As shown in Fig. 3, instead of sampling from the predefined prior distribution, we seek to sample points from a learned latent manifold for training GANs. Specifically, we use an autoencoder (AE) to learn the embeddings over a latent manifold of real data and then employ LCC to learn a set of bases to form local coordinate systems on the latent manifold. After that, we introduce LCC into GANs by approximating the generator using a linear function w.r.t. the set of codings (See Section 4.1). Relying on such an approximation, we propose an LCC-based sampling method to exploit the local information of data (See Section 4.3). The details of our method are illustrated in the following subsections.

### 4.1 Generator Approximation Based on LCC

Based on Definition 3, any point on the latent manifold can be approximated by a linear combination of a set of local bases. Inspired by this, if the bases are sufficiently localized, the generator of GANs can also be approximated by a linear function w.r.t. a set of codings. Therefore, we approximate the generator as follows.

**Lemma 1 (Generator Approximation [2])** Let  $(\gamma, C)$  be an arbitrary coordinate coding on  $\mathbb{R}^{d_B}$ . Given an  $(L_h, L_G)$ -Lipschitz smooth generator  $G_u(\mathbf{h})$ , for all  $\mathbf{h} \in \mathbb{R}^{d_B}$ :

$$\begin{aligned} & \left\| G_u \left( \sum_{v \in C} \gamma_v(\mathbf{h}) v \right) - \sum_{v \in C} \gamma_v(\mathbf{h}) G_u(v) \right\| \\ & \leq 2L_h \|\mathbf{h} - r(\mathbf{h})\| + L_G \sum_{v \in C} |\gamma_v(\mathbf{h})| \cdot \|v - r(\mathbf{h})\|^2, \end{aligned} \quad (2)$$

where  $r(\mathbf{h}) = \sum_{v \in C} \gamma_v(\mathbf{h}) v$ .

In Lemma 1, given the local bases and a Lipschitz smooth generator, the generator w.r.t. the linear combination of the local bases can be approximated by the linear combination of the generator w.r.t. local bases. In general, two close latent points often share the same local bases but with different weights (*i.e.*, local codings), we thus can simply change these weights to approximate the generator. In this way, the pieces of generated data are able to cover the entire manifold seamlessly (See Fig. 2(b)). For convenience, we introduce the following definition to measure the locality of a coding, and then develop our optimization problem.

**Definition 5 (Localization Measure)** Given  $L_h$ ,  $L_G$ , and coding  $(\gamma, C)$ , we define the localization measure  $Q_{L_h, L_G}(\gamma, C)$  as

$$Q_{L_h, L_G}(\gamma, C) = 2L_h \|\mathbf{h} - r(\mathbf{h})\| + L_G \sum_{v \in C} |\gamma_v(\mathbf{h})| \cdot \|v - r(\mathbf{h})\|^2.$$

Following the setting of [23], we set  $L_h=0.25$  and  $L_G=2$  in practice. By minimizing the localization quality, we will propose an objective function of the proposed method.

### 4.2 Objective Function of LCCGAN

Based on the generator approximation method with LCC, we propose a learning method by exploiting LCC coding to train GAN models. Specifically, we first learn the LCC coordinate system. Then, we propose the training objective for the LCCGAN models. The training algorithm is shown in Algorithm 1.

**Learning LCC systems.** In Step 1 of Fig. 3, we show an illustration of how to construct bases to form LCC systems. We first learn an autoencoder to extract the embeddings (*i.e.*, black points) from real data and map them to a latent manifold. Then, based on the extracted embeddings, we seek to use LCC by learning a set of bases to represent the manifold. In this way, any point located on the manifold of embeddings can be represented by the coordinate system constructed using these bases [23].

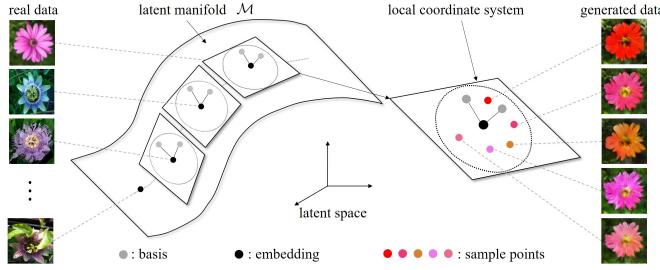


Fig. 4. The geometric views on LCC Sampling. By learning embeddings (*i.e.*, black points) that lie on the latent manifold, we use LCC to learn a set of bases (*i.e.*, gray points) to form a local coordinate system such that we can sample different latent points (*i.e.*, colored points) by LCC sampling. As a result, the proposed LCCGAN-v2 can generate new data that have different attributes.

**Objective of LCC.** To learn the bases (*i.e.*, gray points in Fig. 3), we optimize the objective function of LCC, *i.e.*, we minimize the localization measure to obtain a set of local bases. Specifically, given a set of the latent points  $\{\mathbf{h}_i\}_{i=1}^N$ , by assuming  $\mathbf{h} \approx \mathbf{r}(\mathbf{h})$  [23], we seek to address the following optimization problem:

$$\begin{aligned} & \min_{\gamma, \mathcal{C}} \sum_{\mathbf{h}} 2L_h \|\mathbf{h} - \mathbf{r}(\mathbf{h})\| + L_G \sum_{\mathbf{v} \in \mathcal{C}} |\gamma_{\mathbf{v}}(\mathbf{h})| \cdot \|\mathbf{v} - \mathbf{h}\|^2, \\ & \text{s.t. } \sum_{\mathbf{v} \in \mathcal{C}} \gamma_{\mathbf{v}}(\mathbf{h}) = 1, \forall \mathbf{h}, \end{aligned} \quad (3)$$

where  $\mathbf{h}$  denotes an embedding learned by an autoencoder from real data,  $\mathcal{C}$  denotes the set of local bases, and  $\mathbf{r}(\mathbf{h}) = \sum_{\mathbf{v} \in \mathcal{C}} \gamma_{\mathbf{v}}(\mathbf{h}) \mathbf{v}$ . In practice, we normalize the weights  $\gamma$  to the sum of 1 during the training, and update  $\gamma$  and  $\mathcal{C}$  by alternately optimizing a LASSO problem and a least-square regression problem, respectively. After optimizing Problem (3), we can construct the local bases on the latent manifold.

**Training LCCGAN.** After solving Problem (3), every latent point  $\mathbf{h} \in \mathbb{R}^{d_B}$  would be close to its physical approximation  $\mathbf{r}(\mathbf{h})$ , *i.e.*,  $\mathbf{h} \approx \mathbf{r}(\mathbf{h})$ , then the generator can be approximated by

$$G_u(\mathbf{h}) \approx G_u(\mathbf{r}(\mathbf{h})) \triangleq G_w(\gamma(\mathbf{h})), \mathbf{h} \in \mathcal{H}, \quad (4)$$

where  $\mathbf{r}(\mathbf{h}) = \mathbf{V}\gamma(\mathbf{h})$ ,  $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M]$  and  $\gamma(\mathbf{h}) = [\gamma_1(\mathbf{h}), \gamma_2(\mathbf{h}), \dots, \gamma_M(\mathbf{h})]^T$  with  $M = |\mathcal{C}|$ . Here,  $\mathcal{H}$  is the latent distribution and  $w \in \mathcal{W}$  are the parameters of the generator *w.r.t.*  $u$  and fixed  $\mathbf{V}$  learned from Problem (3). Note that the input of the generator  $G_w(\gamma(\mathbf{h}))$  in this paper is local coordinate coding, which is different from other GAN methods.

According to Definition 4, we apply the neural network distance to measure the divergence between the generated distribution and the empirical distribution. Specifically, given the generator  $G_w(\gamma(\mathbf{h}))$ , we consider optimizing the following objective function for LCCGAN:

$$\min_{G_w \in \mathcal{G}} d_{\mathcal{F}, \phi} \left( \hat{\mathcal{D}}_{G_w}(\gamma(\mathbf{h})), \hat{\mathcal{D}}_{\text{real}} \right), \mathbf{h} \in \mathcal{H}, \quad (5)$$

where  $\mathcal{G}$  is the class of generators,  $\hat{\mathcal{D}}_{G_w}$  is the empirical distribution generated by  $G_w$ , and  $\hat{\mathcal{D}}_{\text{real}}$  is the real distribution. Specifically, Problem (5) can be rewritten as:

$$\min_{w \in \mathcal{W}} \max_{\mathbf{v} \in \mathcal{V}} \mathbb{E}_{\mathbf{x} \sim \hat{\mathcal{D}}_{\text{real}}} [\phi(D_v(\mathbf{x}))] + \mathbb{E}_{\mathbf{h} \sim \mathcal{H}} [\phi(1 - D_v(G_w(\gamma(\mathbf{h})))]) ,$$

where  $\phi(\cdot)$  is a monotone function, and thus the above objective function can be used in different GAN methods, such as DCGAN and WGAN-GP. The detailed algorithm is shown in Algorithm 1.

### Algorithm 1 Training Method for LCCGAN.

---

**Require:** Training data  $\{\mathbf{x}_i\}_{i=1}^N$ ; a prior distribution  $p(\mathbf{z})$ , where  $\mathbf{z} \in \mathbb{R}^d$ ; minibatch size  $n$ ;  $q_{v1} = 2$ ;  $q_{v2} = 3$ .

- 1: Learn the latent manifold  $\mathcal{M}$  using an autoencoder
- 2: Construct LCC bases  $\{\mathbf{v}_i\}_{i=1}^M$  on  $\mathcal{H}$  by optimizing:

$$\min_{\gamma, \mathcal{C}} \sum_{\mathbf{h}} 2L_h \|\mathbf{h} - \mathbf{r}(\mathbf{h})\| + L_G \sum_{\mathbf{v} \in \mathcal{C}} |\gamma_{\mathbf{v}}(\mathbf{h})| \cdot \|\mathbf{v} - \mathbf{h}\|^2$$

- 3: **for** number of training iterations **do**
- 4: Do LCC Sampling to obtain a minibatch  $\{\gamma(\mathbf{h}_i)\}_{i=1}^n$
- 5: Sample a minibatch  $\{\mathbf{x}_i\}_{i=1}^n$  from the data distribution
- 6: Update the discriminator by ascending the gradient:

$$\nabla_{\mathbf{v}} \frac{1}{n} \sum_{i=1}^n \phi(D_v(\mathbf{x}_i)) + \phi((1 - D_v(G_w(\gamma(\mathbf{h}_i))))))$$

- 7: Do LCC Sampling to obtain a minibatch  $\{\gamma(\mathbf{h}_i)\}_{i=1}^n$
- 8: Update the generator by descending the gradient:

$$\nabla_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^n \phi(1 - D_v(G_w(\gamma(\mathbf{h}_i))))$$
- 9: **end for**

---

### 4.3 LCC Sampling Method

To solve Problem (5), one of the key issues is how to conduct sampling from the learned latent manifold. Although the latent manifold can be learned by an autoencoder, it is difficult to sample valid points on it to train GANs. To address this, we propose an LCC sampling method to capture the latent distribution on the learned latent manifold (See Fig. 4). The proposed sampling method contains the following three steps.

**Step 1:** Given a local coordinate system, we construct an  $d_B \times M$  matrix  $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M]$  as the local bases. Here, each basis  $\mathbf{v}_i$  is an  $d_B$ -dimensional vector and  $M$  is the number of bases.

**Step 2:** With the learned local bases  $\mathbf{V}$ , we randomly sample a latent point (specifically, it can be a basis), and then find its  $d$ -nearest neighbors  $\mathcal{B} = \{\mathbf{v}_j\}_{j=1}^d$ .

**Step 3:** To conduct the local sampling method, we construct an  $M$ -dimensional vector  $\gamma(\mathbf{h}) = [\gamma_1(\mathbf{h}), \gamma_2(\mathbf{h}), \dots, \gamma_d(\mathbf{h})]^T$  as the LCC coding. The weight  $\gamma_j(\mathbf{h})$  for the  $j$ -th element of  $\gamma(\mathbf{h})$  can be computed as follows:

$$\gamma_j(\mathbf{h}) = \begin{cases} z_j, & \mathbf{v}_j \in \mathcal{B} \\ 0, & \mathbf{v}_j \notin \mathcal{B} \end{cases}, \quad (6)$$

where  $z_j$  is the  $j$ -th element of  $\mathbf{z} \in \mathbb{R}^d$  from the prior distribution  $p(\mathbf{z})$ . Here, we set  $p(\mathbf{z})$  to be Gaussian distribution  $\mathcal{N}(\mathbf{0}, \mathbf{I})$  and normalize the sum of  $\gamma(\mathbf{h})$  to be 1 in the training, *i.e.*,  $\sum_j \gamma_j(\mathbf{h}) = 1$ . In this paper, we use Gaussian distribution for two reasons. First, Gaussian distribution is an available way for sampling, which has been widely used in many GAN methods [1], [3]. In Fig. 4, given the latent manifold, we employ LCC to form local coordinate systems over the latent manifold, *i.e.*, built with a set of local bases (*i.e.*, gray points). In the local coordinate system, we use Gaussian distribution to sample a new point  $\mathbf{V}\gamma(\mathbf{h})$  (*i.e.*, colored point) by specifying the weights for the local bases. In this way, we can generate images by exploiting the local information of data. Second, by using Gaussian distribution for sampling, it is reasonable and fair to compare LCCGAN with other GAN methods. In Table 1 of Section 6, we are able to justify the advantage of LCCGAN using the local coordinate system over other GANs using the global coordinate system.

Based on Definition 3, the intrinsic dimensionality is determined by the number of bases in a local region. Thus, we turn the determination of the intrinsic dimension into an easier problem of selecting a sufficient number of local bases.

#### 4.4 Relationship between LCC and LCCGAN

In this section, we first discuss the relationship between LCC and the LCC sampling method. Then, we analyze the effect of LCC to improve the performance of GAN models.

**Relationship between LCC and LCC sampling.** The LCC sampling method is closely related to LCC for two reasons. First, both of them rely on the local coordinate system. Specifically, as shown in Fig. 4, we learn a set of bases (*i.e.*, gray points) to form a local coordinate system by optimizing the objective of LCC. Second, both of them can effectively exploit the local information of data. Based on the learned bases, we can use the proposed LCC sampling method to sample different points (*i.e.*, colored points) in a local area of the latent manifold. In practice, within the same local system, the generated samples often share the similar structure or texture.

**How does LCC improve GAN methods?** When introducing LCC into GAN methods, they are able to use the local coordinate system to exploit the local information of data, and thus improve the performance of GANs. In contrast, most GAN methods [1], [3] use a global coordinate system, which, however, would fail to capture the semantic information of data. In this sense, they are possible to sample meaningless points in such global coordinate systems. To verify this, the advantage of the local coordinate system over the global coordinate system has been demonstrated in Table 1.

#### 4.5 Theoretical Analysis

We first provide some necessary notation to develop our theoretical analysis for LCC based GANs. Let  $\{\mathbf{x}_i\}_{i=1}^N$  be a set of observed training samples drawn from the real distribution  $\mathcal{D}_{\text{real}}$ , and let  $\widehat{\mathcal{D}}_{\text{real}}$  denote the empirical distribution over  $\{\mathbf{x}_i\}_{i=1}^N$ . Given a generator  $G_u$  and a set of the latent points  $\{\mathbf{h}_i\}_{i=1}^r$ ,  $\{G_u(\mathbf{h}_i)\}_{i=1}^r$  denotes a set of  $r$  generated samples from the generated distribution  $\mathcal{D}_{G_u}$ , and  $\widehat{\mathcal{D}}_{G_u}$  is an empirical generated distribution. Motivated by [32], [33], we define the generalization of LCCGAN-v1 as follows:

**Definition 6 (Generalization)** *The neural network distance  $d_{\mathcal{F}, \phi}(\cdot, \cdot)$  between distributions generalizes with  $N$  training samples and error  $\epsilon$ , if for a learned distribution  $\mathcal{D}_{G_u}$ , the following holds with high probability,*

$$\left| d_{\mathcal{F}, \phi} \left( \widehat{\mathcal{D}}_{G_u}, \mathcal{D}_{\text{real}} \right) - \inf_{\mathcal{G}} d_{\mathcal{F}, \phi} \left( \mathcal{D}_{G_u}, \mathcal{D}_{\text{real}} \right) \right| \leq \epsilon.$$

In Definition 6, the generalization of GANs means that the population distance  $d_{\mathcal{F}, \phi}(\mathcal{D}_{G_u}, \mathcal{D}_{\text{real}})$  is close to the distance  $d_{\mathcal{F}, \phi}(\widehat{\mathcal{D}}_{G_u}, \mathcal{D}_{\text{real}})$ . In theory, we hope to obtain a small  $d_{\mathcal{F}, \phi}(\mathcal{D}_{G_u}, \mathcal{D}_{\text{real}})$ . In practice, we can minimize the empirical loss  $d_{\mathcal{F}, \phi}(\widehat{\mathcal{D}}_{G_u}, \widehat{\mathcal{D}}_{\text{real}})$  to approximate  $d_{\mathcal{F}, \phi}(\widehat{\mathcal{D}}_{G_u}, \mathcal{D}_{\text{real}})$ .

To analyze the generalization analysis of LCCGAN-v2, we first give the following relevant lemma. When the latent points lie on a latent manifold and the generator is Lipschitz smooth,  $Q_{L_h, L_G}(\gamma, \mathcal{C})$  has a bound as follows.

**Lemma 2 (Manifold Coding [23])** *If the latent points lie on a compact smooth manifold  $\mathcal{M}$ , given an  $(L_h, L_G)$ -Lipschitz smooth generator  $G_u(\mathbf{h})$  and any  $\epsilon > 0$ , then there exist anchor points  $\mathcal{C} \subset \mathcal{M}$  and coding  $\gamma$  such that*

$$Q_{L_h, L_G}(\gamma, \mathcal{C}) \leq \left[ 2L_h c_{\mathcal{M}} + \left( 1 + \sqrt{d_{\mathcal{M}}} + 4\sqrt{d_{\mathcal{M}}} \right) L_G \right] \epsilon^2,$$

where  $d_{\mathcal{M}}$  is the dimension of the latent manifold.

In Lemma 2, the complexity of local coordinate coding depends on the intrinsic dimension of the manifold instead of the dimension of the basis. Then, we have the following generalization bound on  $\widehat{\mathcal{D}}_{\text{real}}$  to develop the generalization analysis of LCCGAN-v1.

**Theorem 1** *Suppose that  $\phi(\cdot)$  is Lipschitz smooth:  $|\phi'(\cdot)| \leq L_\phi$ , and bounded in  $[-\Delta, \Delta]$ . Given the coordinate coding  $(\gamma, \mathcal{C})$ , an example set  $\mathcal{H}$  in latent space and the empirical distribution  $\widehat{\mathcal{D}}_{\text{real}}$ , if the generator is Lipschitz smooth, then the expected generalization error satisfies:*

$$\begin{aligned} & \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \widehat{\mathcal{D}}_{G_{\widehat{w}}}(\gamma(\mathbf{h})), \widehat{\mathcal{D}}_{\text{real}} \right) \right] \\ & \leq \inf_{\mathcal{G}} \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \mathcal{D}_{G_u}(\mathbf{h}), \widehat{\mathcal{D}}_{\text{real}} \right) \right] + \epsilon(d_{\mathcal{M}}), \end{aligned}$$

$$\text{where } \epsilon(d_{\mathcal{M}}) = L_\phi Q_{L_h, L_G}(\gamma, \mathcal{C}) + 2\Delta.$$

**Proof** See supplementary materials for the proof.  $\square$

Theorem 1 shows that the generalization bound for  $\widehat{\mathcal{D}}_{\text{real}}$  is related to the dimension of the latent manifold (*i.e.*,  $d_{\mathcal{M}}$ ) rather than the dimension of the latent space (*i.e.*,  $d_B$ ). Based on Theorem 1 and the Rademacher complexity [36], we then accomplish the generalization bound on an unknown real distribution  $\mathcal{D}_{\text{real}}$ .

**Theorem 2** *Under the condition of Theorem 1, given an empirical distribution  $\widehat{\mathcal{D}}_{\text{real}}$  with  $N$  samples drawn from  $\mathcal{D}_{\text{real}}$ , the following holds with probability at least  $1 - \delta$ ,*

$$\begin{aligned} & \left| \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \widehat{\mathcal{D}}_{G_{\widehat{w}}}, \mathcal{D}_{\text{real}} \right) \right] - \inf_{\mathcal{G}} \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \mathcal{D}_{G_u}, \mathcal{D}_{\text{real}} \right) \right] \right| \\ & \leq 2R_{\mathcal{X}}(\mathcal{F}) + 2\Delta \sqrt{\frac{2}{N} \log \left( \frac{1}{\delta} \right)} + 2\epsilon(d_{\mathcal{M}}), \end{aligned}$$

where  $R_{\mathcal{X}}(\mathcal{F})$  is the Rademacher complexity of  $\mathcal{F}$ .

**Proof** See supplementary material for the proof.  $\square$

Theorem 2 shows that the generalization error of LCCGAN-v1 can be bounded by the Rademacher complexity of  $\mathcal{F}$  and an error term  $\epsilon(d_{\mathcal{M}})$ . Specifically, the former term  $R_{\mathcal{X}}(\mathcal{F})$  implies that the set of discriminators should be smaller to have better generalization ability, and be large enough to be able to identify the data distribution, which is consistent with [33]. The latter term  $\epsilon(d_{\mathcal{M}})$  indicates that a low dimensional input is sufficient to achieve good generalization. In practice, every dataset has its own dimension of the latent manifold. Nevertheless, experiments show that the proposed method is able to generate perceptually convincing images with small-dimensional inputs.

#### 5 IMPROVED ADVERSARIAL LEARNING WITH LCC

Based on the LCCGAN-v1 method, we propose an enhanced GAN method, called LCCGAN-v2, to improve the approximation of the generator. Specifically, we first introduce a higher-order term to approximate the generator of GANs in Section 5.1. We further analyze the generalization performance and propose a new corollary to prove that the higher-order term is able to obtain better generator approximation, and thus yield better performance (See the results in Section 6).

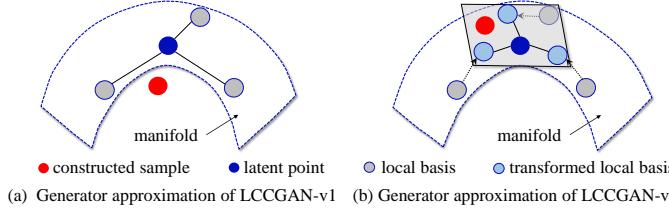


Fig. 5. Comparisons of the LCC sampling between LCCGAN-v1 and LCCGAN-v2. (a) For LCCGAN-v1, when the number of the local bases is insufficient, the approximation of the generator would not be accurate. As a result, the constructed sample would be far away from the manifold. (b) For LCCGAN-v2, we approximate the generator in a locally flat region, and thus the constructed sample is close to the manifold.

### 5.1 Extended Generator Approximation

By minimizing the right-hand side of (2), the generator equipped with LCC-v1 [2] has a small approximation error. However, the local linear approximation may not necessarily be optimal when the generator is highly nonlinear. It means that many local bases are required to achieve better approximation. As suggested by [35], the higher-order error term would have a better generator approximation. Thus, we can improve the LCC-v1 method by introducing a higher-order correction term. For convenience, we produce the improved LCC-v2 and show the corresponding extended generator approximation in the following lemma.

**Lemma 3 (Extended Generator Approximation)** Let  $(\gamma, \mathcal{C})$  be an arbitrary coordinate coding on  $\mathbb{R}^{d_B}$ . Given an  $(L_h, L_\nu)$ -Lipschitz smooth generator  $G_u(\mathbf{h})$ , for all  $\mathbf{h} \in \mathbb{R}^{d_B}$ :

$$\left\| G_u(\mathbf{r}(\mathbf{h})) - \sum_{\mathbf{v} \in \mathcal{C}} \gamma_{\mathbf{v}}(\mathbf{h}) \left( G_u(\mathbf{v}) + \frac{1}{2} \nabla G_u(\mathbf{v})^\top (\mathbf{h} - \mathbf{v}) \right) \right\| \leq 2L_h \|\mathbf{h} - \mathbf{r}(\mathbf{h})\| + L_\nu \sum_{\mathbf{v} \in \mathcal{C}} |\gamma_{\mathbf{v}}(\mathbf{h})| \cdot \|\mathbf{v} - \mathbf{r}(\mathbf{h})\|^3, \quad (7)$$

where  $\mathbf{r}(\mathbf{h}) = \sum_{\mathbf{v} \in \mathcal{C}} \gamma_{\mathbf{v}}(\mathbf{h}) \mathbf{v}$ .

**Proof** See supplementary material for the proof.  $\square$

In Lemma 3, the generator w.r.t. the linear combination of the local bases can be approximated by introducing gradient directions. Based on LCC-v1, we improve LCC by using a higher order term, called LCC-v2. Compared the right-hand side of (2) with (7), the first term is similar and can be small when  $\mathbf{h}$  can be well approximated by a linear combination of local bases, which happens when the manifold is relatively flat. For the second term, the local linear approximation measured by  $L_G$  is replaced by the local quadratic approximation measured by  $L_\nu$ . For convenience, we let the right-hand side of Eqn. (7) be  $Q_{L_h, L_\nu}(\gamma, \mathcal{C})$ . Then, we slightly extend Lemma 2 to the following lemma:

**Lemma 4** If the latent points lie on a compact smooth manifold  $\mathcal{M}$ , given an  $(L_h, L_\nu)$ -Lipschitz smooth generator  $G_u(\mathbf{h})$  and any  $\epsilon > 0$ , then there exist anchor points  $\mathcal{C} \subset \mathcal{M}$  and coding  $\gamma$  such that

$$Q_{L_h, L_\nu}(\gamma, \mathcal{C}) \leq \left[ 2L_h c_{\mathcal{M}} + \left( 1 + \sqrt{d_{\mathcal{M}}} + 8\sqrt{d_{\mathcal{M}}} \right) L_\nu \right] \epsilon^3,$$

where  $d_{\mathcal{M}}$  is the dimension of the latent manifold.

In Lemma 4, the complexity of the local coordinate coding depends on the intrinsic dimension of the latent manifold instead of the dimension of the basis.

### 5.2 Differences between LCCGAN-v1 and LCCGAN-v2

We demonstrate the differences between these two methods in Fig. 5. For LCCGAN-v1, when the number of the local bases is insufficient, the approximation of the generator would not be accurate. As a result, the constructed sample would be far away from the manifold. For LCCGAN-v2, the generator w.r.t. the local bases can be transformed into the locally flat region approximately along the gradient of the generator. In this way, the linear combination of the generator w.r.t. the local bases would be close to the manifold. Therefore, with the linear combination of bases as input, we have a good generator approximation to generate realistic images.

### 5.3 Further Generalization Analysis

To further analyze the generalization analysis of LCCGAN-v2, we first give the following relevant lemma.

**Theorem 3** Under the condition of Theorem 1, given an empirical distribution  $\hat{\mathcal{D}}_{\text{real}}$  with  $N$  samples drawn from  $\mathcal{D}_{\text{real}}$ , the following holds with probability at least  $1 - \delta$ ,

$$\begin{aligned} & \left| \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \hat{\mathcal{D}}_{G_{\hat{w}}}, \mathcal{D}_{\text{real}} \right) \right] - \inf_{\mathcal{G}} \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \mathcal{D}_{G_u}, \mathcal{D}_{\text{real}} \right) \right] \right| \\ & \leq 2R_{\mathcal{X}}(\mathcal{F}) + 2\Delta \sqrt{\frac{2}{N} \log \left( \frac{1}{\delta} \right)} + 2\epsilon(d_{\mathcal{M}}), \end{aligned}$$

where  $R_{\mathcal{X}}(\mathcal{F})$  is the Rademacher complexity of  $\mathcal{F}$  and

$$\epsilon(d_{\mathcal{M}}) = L_\phi Q_{L_h, L_\nu}(\gamma, \mathcal{C}) + 2\Delta.$$

**Proof** See supplementary materials for the proof.  $\square$

Note that the theorem is slightly different from [2] because the localization quality is related to the high-order term. Based on Theorem 3 and the Rademacher complexity [36], we consider a specific discriminator set and apply Theorem 2 to analyze and understand the generalization performance of LCCGAN-v2.

**Corollary 1** Let  $\mathcal{X}$  be the unit ball of  $\mathbb{R}^d$  under the  $\ell_2$ -norm, i.e.,  $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \leq 1\}$ . Assume that the discriminator set  $\mathcal{F}$  is the set of neural networks with a rectified linear unit (ReLU),

$$\mathcal{F} = \left\{ \max \{ \mathbf{w}^\top [\mathbf{x}; 1], 0 \} : \mathbf{w} \in \mathbb{R}^{d+1}, \|\mathbf{w}\| = 1 \right\},$$

then with probability at least  $1 - \delta$ ,

$$\begin{aligned} & \left| \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \hat{\mathcal{D}}_{G_{\hat{w}}}, \mathcal{D}_{\text{real}} \right) \right] - \inf_{\mathcal{G}} \mathbb{E}_{\mathcal{H}} \left[ d_{\mathcal{F}, \phi} \left( \mathcal{D}_{G_u}, \mathcal{D}_{\text{real}} \right) \right] \right| \\ & \leq 2\Delta \sqrt{\frac{2}{N} \log \left( \frac{1}{\delta} \right)} + \frac{4\sqrt{2}}{\sqrt{N}} + 2\epsilon(d_{\mathcal{M}}). \end{aligned}$$

**Proof** See supplementary materials for the proof.  $\square$

In Corollary 1, using a one-layered ReLU network, the generalization bound of the proposed GAN method is related to the error term w.r.t. the dimension of the latent distribution. In other words, with a low dimensional input and sufficient training data, LCCGAN-v2 is able to obtain better generator approximation, and thus achieve better generalization performance in practice.

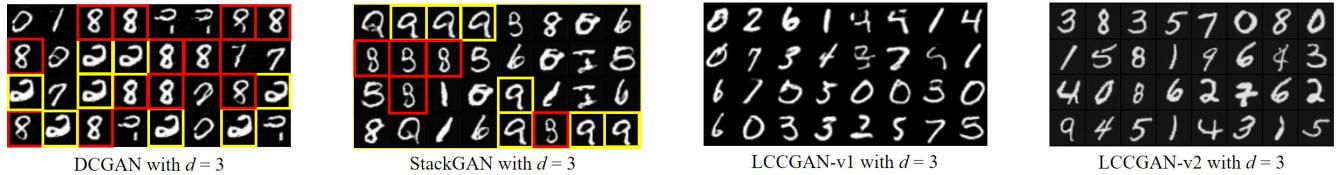


Fig. 6. Generated samples with  $d = 3$ . The yellow and red boxes denote the similar generated digits with low diversity.

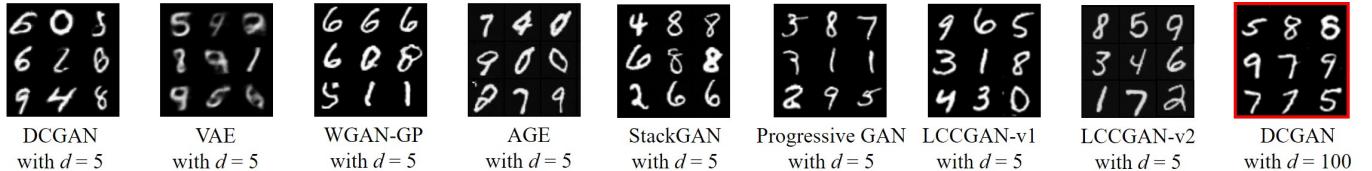


Fig. 7. Comparison with different GAN methods with the input noise of  $d = 5$ . DCGAN with  $d = 100$  is considered as the baseline.

## 6 EXPERIMENTS

We evaluate the proposed method on a variety of real-world datasets including MNIST [37], Oxford-102 [38], LSUN [39] and CelebA [40]. For all considered GAN methods, the inputs are sampled from a  $d$ -dimensional prior distribution, and we train the generative models to produce  $64 \times 64$  images. Considering the reproducibility, we have made the code for both LCCGAN-v1<sup>1</sup> and LCCGAN-v2<sup>2</sup> available on the internet.

We organize the experiments as follows. First, we introduce some details about the benchmark datasets and the evaluation metrics in Section 6.1. Then, we give the implementation details in Section 6.2. Finally, we compare our method with several baseline methods on four benchmark datasets in Section 6.3.

### 6.1 Datasets and Evaluation Metrics

To comprehensively evaluate the proposed method, we conduct experiments on a wide variety of datasets including:

- **MNIST** [37] contains 60,000 handwritten digit images ranging from 0 to 9. For each digit, it has been size-normalized and centered in a  $28 \times 28$  image.
- **Oxford-102** [38] has 8,189 flower images of 102 different categories. We normalize each image and resize it into  $64 \times 64$ .
- **LSUN** [39] has 10 scene categories, and each category consists of approximately 1,000,000 natural images of indoor scenes. We select the bedroom category and the classroom category to train the generative models.
- **CelebA** [40] consists of a group of celebrity faces. Since these images often share a common face outline and only differ in detailed attributes, e.g., hair, eyes, mouth, and skin, we train the models with a larger dimension to capture the local information.

For quantitative evaluation, we use two widely used metrics, *Inception Score* (IS) [43] and *Fréchet Inception Distance* (FID) [44], to evaluate the generated samples. Specifically, IS measures both the single image quality and the diversity over a large number of samples (*i.e.*, 50k), and a larger IS value corresponds to the better performance of the method. FID captures the similarity between real and generated images, and a smaller FID value indicates the better performance. Note that both metrics are highly consistent with human evaluations.

1. <https://github.com/guoyongcs/LCCGAN>.  
2. <https://github.com/SCUTjinchengli/LCCGAN-v2>.

### 6.2 Implementation Details

We compare our method with several baseline methods, including DCGAN [17], VAE [26], WGAN-GP [24], AGE [41], StackGAN [42], Progressive GAN [18] and LCCGAN-v1 [2]. Note that StackGAN is originally devised with an input text as the condition. However, since there is no text data acting as the condition in our experiments, we remove its module of text embedding.

In the training, we follow the experimental settings in DCGAN [17]. Specifically, we use Adam [45] with a mini-batch size of 64 and a learning rate of 0.0002 to train the generator and the discriminator. Following the strategy in [46], we initialize the parameters of both the generator and the discriminator. We set the hyperparameters  $L_h=1$  and  $L_v=0.0001$ . All experiments are conducted on a single NVIDIA Titan X GPU.

### 6.3 Comparison Results

#### 6.3.1 Comparisons on MNIST

In this experiment, we compare different GAN methods on MNIST and show the visual comparisons in Figs. 6 and 7. From Fig. 6, given a very low dimensional input with  $d = 3$ , DCGAN and StackGAN produce only few kinds of digits with almost the same shapes (See the yellow and red boxes in Fig. 6). However, LCCGAN-v1 with  $d = 3$  often produces digits with different styles and different orientations. Furthermore, with the  $d = 3$  input, LCCGAN-v2 further produces images with better visual fidelity and higher diversity. Equipped with LCC, the proposed method effectively preserves the local information of data on the latent manifold and thus helps the training of GANs.

In Fig. 7, we increase the dimension of input to  $d = 5$  and compare the proposed method with other GAN methods. In this experiment, the considered baseline methods often produce the digits with distorted structures. In contrast, with such a low dimensional input, LCCGAN-v1 is able to produce the images with meaningful content. Furthermore, LCCGAN-v2 significantly outperforms the considered baseline methods and produces sharper images associated with higher diversity. More critically, with the help of LCC coding, LCCGAN-v1 and LCCGAN-v2 with  $d = 5$  input are able to achieve comparable or even better performance than their GAN counterparts with  $d = 100$  (See the red box in Fig. 7). These results show the effectiveness of the proposed method in training generative models by exploiting the local information of the latent manifold.

TABLE 1  
Comparison with different GAN methods in terms of Inception-Score (IS) and Fréchet Inception Distance (FID) on Oxford-102.

Methods	$d = 5$		$d = 10$		$d = 30$		$d = 100$	
	IS	FID	IS	FID	IS	FID	IS	FID
DCGAN [17]	$2.355 \pm 0.019$	187.5	$3.262 \pm 0.022$	204.7	$3.050 \pm 0.015$	186.2	$2.683 \pm 0.022$	182.2
VAE [26]	$2.451 \pm 0.018$	245.6	$2.358 \pm 0.022$	190.6	$2.234 \pm 0.016$	244.0	$2.856 \pm 0.024$	214.8
WGAN-GP [24]	$2.719 \pm 0.031$	185.2	$2.891 \pm 0.025$	179.8	$3.081 \pm 0.018$	136.7	$3.458 \pm 0.028$	160.4
AGE [41]	$2.865 \pm 0.024$	234.1	$3.062 \pm 0.021$	186.7	$2.630 \pm 0.023$	211.8	$2.488 \pm 0.014$	235.9
StackGAN [42]	$2.664 \pm 0.013$	164.2	$2.702 \pm 0.015$	167.7	$3.109 \pm 0.018$	197.0	$2.741 \pm 0.022$	178.8
Progressive GAN [18]	$2.844 \pm 0.031$	128.6	$3.295 \pm 0.028$	128.6	$3.196 \pm 0.028$	106.8	$3.532 \pm 0.028$	114.5
LCCGAN-v1 [2]	$3.079 \pm 0.026$	71.2	$3.077 \pm 0.033$	82.7	$3.003 \pm 0.030$	61.9	$3.147 \pm 0.038$	66.7
LCCGAN-v2	<b><math>3.267 \pm 0.023</math></b>	<b>71.0</b>	<b><math>3.394 \pm 0.019</math></b>	<b>71.1</b>	<b><math>3.370 \pm 0.031</math></b>	<b>57.7</b>	<b><math>3.590 \pm 0.020</math></b>	<b>60.7</b>

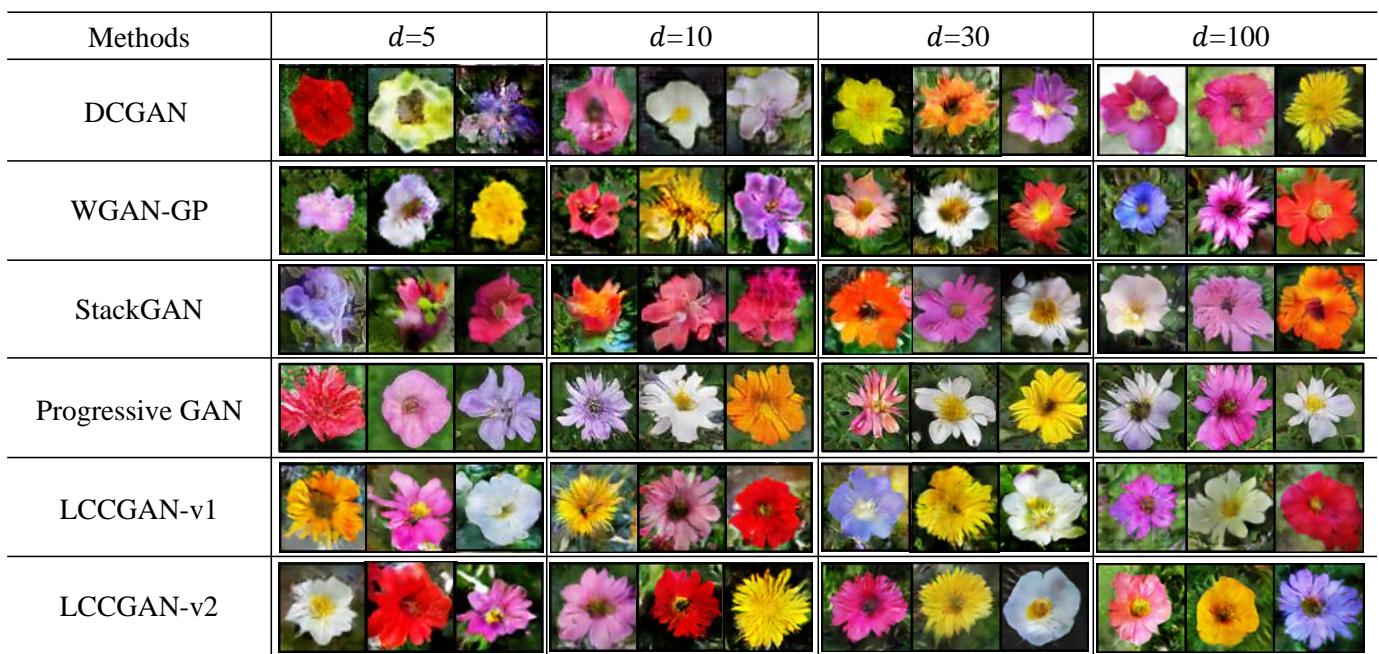


Fig. 8. Visual comparison of GAN methods with different dimensions of the latent distribution on Oxford-102.

### 6.3.2 Comparisons on Oxford-102

We further evaluate the proposed method on a larger dataset Oxford-102. In this experiment, we adjust the input with different dimensions, *e.g.*,  $d = \{5, 10, 30, 100\}$ . We show the qualitative and quantitative results in Fig. 8 and Table 1, respectively.

From Fig. 8, we have the following observations. First, the performance of the baselines highly depends on the input dimension. For example, given a low dimension with  $d=5$  or  $d=10$ , DCGAN often generates images with a blurred structure and distorted regions. In contrast, LCCGAN-v1 is able to produce promising images with a clear structure given an input with  $d=5$ . Moreover, our LCCGAN-v2 often shows better performance than all methods above and produces perceptually convincing images. With such a low dimensional input, the proposed LCCGAN-v1 and LCCGAN-v2 effectively capture the local information of the latent manifold and produce realistic images. Second, we further investigate the effect of the input dimension on the quality of the generated images. From Fig. 8, as the input dimension increases from  $d=5$  to  $d=100$ , LCCGAN-v2 consistently outperforms LCCGAN-v1 and the considered baselines.

Moreover, we also compare the quantitative results of different GAN methods. In particular, Progressive GAN has to be trained with a very large number of iterations and takes about 20 days for the training (reported in the original paper). However, the other GAN methods are only trained with a limited number of iterations to converge and take several hours for the training. In this sense, it is unfair to directly compare different GAN methods with different training settings. To address this issue, we train different GAN models with the same number of iterations to conduct a fair comparison (See implementation details in Subsection 6.2).

From Table 1, given  $d=5$ , Progressive GAN obtains slightly better IS and FID than other methods. In contrast, LCCGAN-v1 and LCCGAN-v2 significantly outperform the other methods with various  $d$  in terms of both IS and FID. More critically, the proposed LCCGAN-v2 with  $d=5$  achieves even better performance than all baselines with  $d=30$  and several methods with  $d=100$ , *e.g.*, DCGAN. It means that LCCGAN only requires a small-dimensional input to achieve good performance. These results show the effectiveness of the proposed method in producing perceptually promising images with high quality and large diversity.

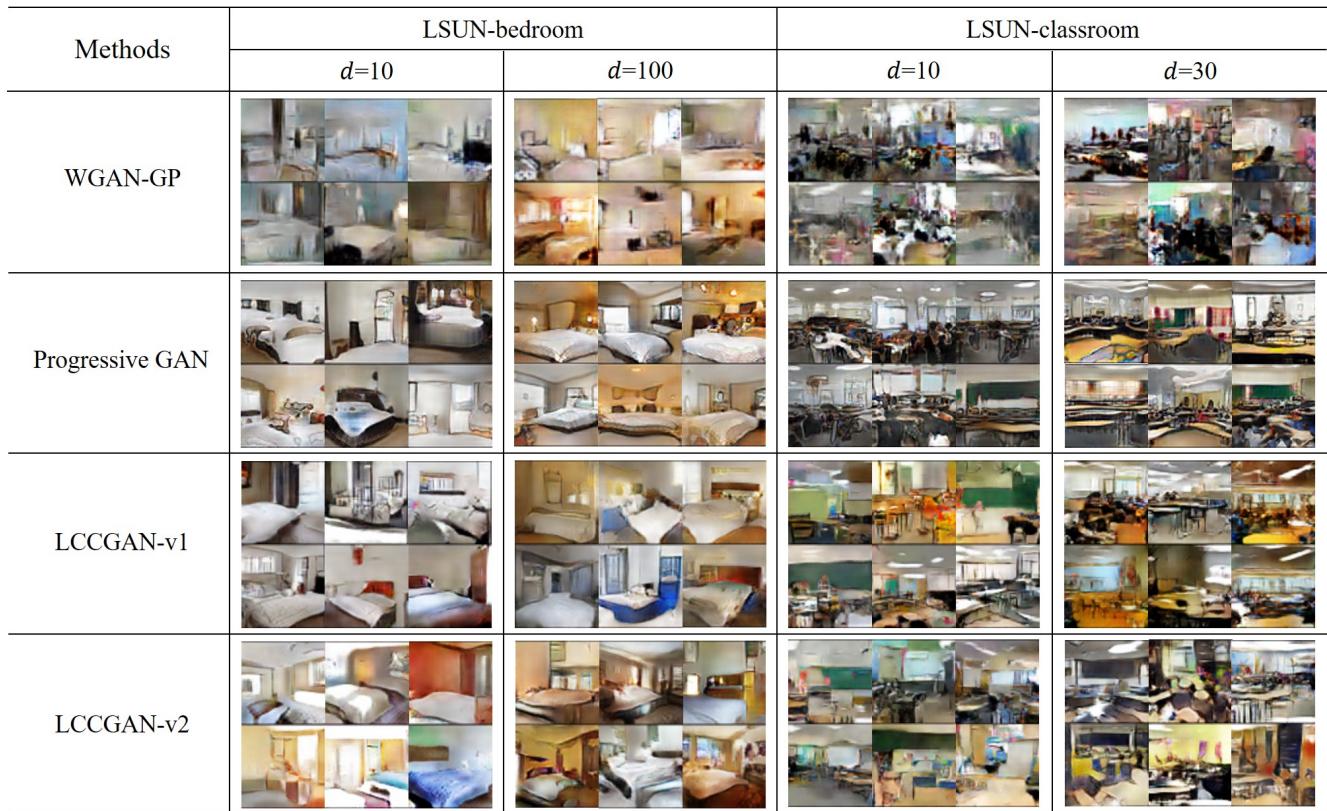
TABLE 2

Comparison with different GAN methods with different dimensions of the latent distribution in terms of Inception-Score (IS) and Fréchet Inception Distance (FID) on the LSUN dataset.

Methods	LSUN-bedroom								LSUN-classroom							
	d = 5		d = 10		d = 30		d = 100		d = 5		d = 10		d = 30		d = 100	
	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID
DCGAN [17]	1.969	253.7	2.531	193.9	2.409	204.6	2.165	239.7	2.230	272.2	2.204	258.8	2.401	233.1	2.347	271.9
VAE [26]	2.785	198.7	2.967	183.3	3.218	166.3	3.265	178.9	2.195	232.7	2.491	164.0	2.646	182.4	2.740	175.4
WGANGP [24]	2.875	172.4	2.834	176.3	2.950	154.2	2.965	172.6	2.595	195.7	2.733	197.6	2.799	169.7	2.701	173.3
AGE [41]	2.031	312.1	2.345	193.8	2.186	219.3	2.602	171.6	2.002	311.0	2.142	267.3	2.278	262.7	1.956	321.5
StackGAN [42]	2.722	237.3	2.637	197.3	2.675	164.5	2.612	238.0	2.292	209.7	1.961	239.0	2.340	256.2	1.855	257.0
Progressive GAN [18]	3.405	161.4	<b>3.763</b>	156.7	<b>3.951</b>	149.3	3.837	154.3	2.673	189.2	3.073	174.9	<b>3.367</b>	170.9	3.176	177.8
LCCGAN-v1 [2]	3.254	104.1	3.213	110.3	3.084	139.1	3.350	115.0	2.786	105.3	<b>3.094</b>	103.0	2.974	103.4	2.532	132.2
LCCGAN-v2	<b>3.406</b>	<b>98.0</b>	3.683	<b>109.8</b>	3.546	<b>88.1</b>	<b>4.109</b>	<b>110.7</b>	<b>2.866</b>	<b>95.2</b>	3.005	<b>96.6</b>	3.201	<b>102.9</b>	<b>3.273</b>	<b>98.9</b>

TABLE 3

Visual comparison of different GAN methods with different input dimensions on the LSUN-bedroom and LSUN-classroom dataset.



### 6.3.3 Comparisons on LSUN

We also conduct experiments on the LSUN dataset [39] to evaluate the performance of our proposed method. Specifically, we compare the proposed LCCGAN-v1 and LCCGAN-v2 with the considered baseline methods. The quantitative and qualitative results in Table 2 and Table 3. Besides the IS metric, we also use FID to evaluate the diversity of the generated images in each class.

From Table 3, given a low dimension of the input ( $d = 10$ ) on the LSUN-bedroom dataset, WGAN-GP and Progressive GAN fail to produce meaningful bedroom images. However, LCCGAN-v1 and LCCGAN-v2 with this lower dimension of the input are able to produce even better images than WGAN-GP and Progressive GAN with  $d=100$ , which often require a large dimensional input.

In Table 2, although Progressive GAN achieves a good inception score with  $d=10$  on the LSUN-bedroom dataset, the proposed LCCGAN-v2 achieves the lower FID score and outperforms Progressive GAN by approximately 46.9. Therefore, with the help of the LCC sampling method, both LCCGAN-v1 and LCCGAN-v2 consistently outperform the considered baseline methods in terms of FID. For the qualitative comparison, as shown in the second column of Table 3, LCCGAN-v1 and LCCGAN-v2 methods are able to produce images with sharper structures and richer details. We also show the generated images on LSUN-classroom in the last two columns of Table 3. Similarly, LCCGAN-v1 and LCCGAN-v2 consistently outperform the considered baselines and produce images with higher fidelity.

TABLE 4  
Visual comparison of GAN methods with different dimensions of the latent distribution on CelebA.

Methods	$d=30$						$d=100$					
DCGAN												
Progressive GAN												
LCCGAN-v1												
LCCGAN-v2												

TABLE 5  
Comparison with different GAN methods in terms of Inception Score (IS) and Fréchet Inception Distance (FID) on CelebA.

Methods	$d = 30$		$d = 100$	
	IS	FID	IS	FID
DCGAN [17]	$2.299 \pm 0.014$	67.2	$2.214 \pm 0.022$	78.5
VAE [26]	$2.395 \pm 0.017$	52.0	$2.308 \pm 0.019$	54.4
WGAN-GP [24]	$2.344 \pm 0.025$	92.0	$2.388 \pm 0.023$	88.9
AGE [41]	$2.517 \pm 0.025$	82.2	$2.612 \pm 0.026$	63.0
StackGAN [42]	$2.036 \pm 0.016$	131.0	$2.419 \pm 0.014$	133.8
Progressive GAN [18]	$2.527 \pm 0.020$	52.8	$2.530 \pm 0.017$	55.2
LCCGAN-v1 [2]	$2.420 \pm 0.027$	54.4	$2.526 \pm 0.025$	31.9
LCCGAN-v2	<b><math>2.582 \pm 0.018</math></b>	<b>29.2</b>	<b><math>2.625 \pm 0.017</math></b>	<b>25.9</b>

### 6.3.4 Comparisons on CelebA

We also conduct experiments on CelebA [40]. Due to the difficulty of producing face images, we use larger input (*e.g.*,  $d=30$ ) to train the generative models in this experiment.

For the qualitative results, Table 4 shows that given the same input dimension, our LCCGAN-v2 shows better performance than LCCGAN-v1 and other baselines. Moreover, by introducing LCC sampling into the training, LCCGAN-v2 with a low input  $d = 30$  produces promising face images with better quality and larger diversity than DCGAN and Progressive GAN with  $d = 100$ .

For the quantitative results, Table 5 shows that LCCGAN-v2 consistently outperforms LCCGAN-v1 and other methods with various  $d$  in terms of both IS and FID scores. Given an input  $d = 30$ , LCCGAN-v2 achieves the best quantitative results with an FID score of 29.2. According to these results, LCCGAN-v2 greatly benefits from the sampling method and makes the training much easier than directly matching the standard Gaussian distribution.

TABLE 6  
Visual comparison of different GAN methods on the ImageNet dataset, including Promontory and Volcano.

Methods	ImageNet-Volcano				ImageNet-Promontory			
DCGAN ( $d = 100$ )								
StackGAN-v1 ( $d=100$ )								
Progressive GAN ( $d=100$ )								
LCCGAN-v2 ( $d=30$ )								

### 6.4 Comparisons on ImageNet

In this experiment, we further evaluate the performance of the proposed LCCGAN method on the ImageNet dataset. Specifically, since we focus on unconditional GAN models in this paper, training 1000 models on ImageNet (1000 categories in total) is infeasible and impractical. Following [19], [47], we choose two of them, *i.e.*, Promontory and Volcano, to compare the performance of different GAN models, including DCGAN, StackGAN and Progressive GAN. We show the visual results in Table 6.

From Table 6, with a small-dimensional input  $d=30$ , our proposed LCCGAN is able to produce promising images with better quality than the considered methods with a high dimension of  $d=100$  on the ImageNet dataset. Therefore, these results demonstrate the effectiveness of our proposed LCCGAN with a low dimension of the input.

TABLE 7  
Effect of the LCC training method on improving the performance of different GAN methods on Oxford-102.

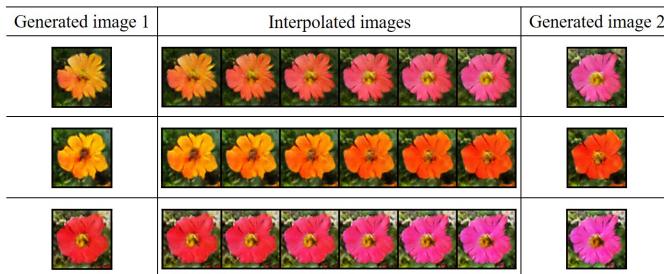
Method	DCGAN		WGAN-GP		StackGAN-v1		StackGAN-v2		Progressive GAN	
	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID
Baseline	2.683 ± 0.022	182.2	3.458 ± 0.028	160.4	2.741 ± 0.022	178.8	3.087 ± 0.027	27.0	3.532 ± 0.028	114.5
with LCC-v1	3.003 ± 0.030	61.9	3.496 ± 0.032	155.5	2.895 ± 0.017	177.6	3.088 ± 0.031	23.7	3.571 ± 0.024	111.2
with LCC-v2	<b>3.370 ± 0.031</b>	<b>57.7</b>	<b>3.546 ± 0.032</b>	<b>145.9</b>	<b>3.005 ± 0.014</b>	<b>168.2</b>	<b>3.216 ± 0.030</b>	<b>22.2</b>	<b>3.710 ± 0.036</b>	<b>109.6</b>

TABLE 8  
Generated images from LCC sampling on MNIST, Oxford-102 and CelebA. The second column shows the images generated from the synthesized latent points. In the last column, we use the Pearson distance to find the closest image in the training data.

Datasets	Generated samples										Closest real data
MNIST											
Oxford-102											
CelebA											

TABLE 9

Interpolations between two generated images on Oxford-102. The first and the last column show the generated images, and the middle column is the interpolated images between two corresponding images.



## 6.5 Effectiveness of the LCCGAN Framework

In this experiment, we verify the effectiveness of the LCCGAN framework by introducing LCC into different GAN methods, including DCGAN, WGAN-GP, StackGAN-v1, StackGAN-v2 and Progressive GAN. Note that we build our LCCGAN based on the DCGAN model (with 3.6M parameters) which is much smaller than StackGAN-v2 (with 16.5M parameters) and Progressive GAN (with 60.7M parameters). In this sense, due to the non-negligible gap in model size, it seems unfair to directly compare the LCC based DCGAN with larger GAN models, like StackGAN-v2 and Progressive GAN. To address this issue, we verify the effectiveness of the proposed LCC method by applying our method to other GAN methods. In this way, we can compare the performance of the models equipped with and without LCC sampling. We show the results in Table 7.

With the help of LCC learning scheme, the resultant models trained with both LCC-v1 and LCC-v2 consistently outperform the baseline models in terms of IS and FID given different dimensions of the input. These results demonstrate the effectiveness of our LCC learning scheme in training GANs.

## 7 ADDITIONAL EXPERIMENTS

In this section, we conduct further analyses and discussions of our proposed method. First, we conduct the LCC sampling and the latent manifold interpolation on the Oxford-102 dataset. Then, we investigate the performance of the proposed LCCGAN using local and entire bases, and the impact of the end-to-end training method. Last, we conduct many ablation studies to demonstrate the effectiveness of the proposed LCCGAN.

### 7.1 Demonstration of the LCC Sampling

In this experiment, we investigate the effectiveness of the LCC sampling. In this way, we are able to verify that the proposed LCCGAN does not memorize the training data by finding the closest (or the most similar) image from the training set and showing the differences from the generated images. Specifically, we first randomly select one latent point in the coordinate system and find the nearest  $d$  bases. Then, we generate 10 latent points using random weights based on the selected  $d$  bases to produce images. Here, we limit the nearest neighbor bases since we use the local coordinate system to generate samples over the latent manifold. By using the nearest neighbor bases, we are able to sample meaningful points to exploit the local information of real data and improve the performance of GAN methods. Ideally, these images should be located in a local area of the latent manifold and share common features.

To verify this, we conduct experiments on several benchmark datasets. From Table 8, the proposed method is able to produce a set of similar images with different orientations or styles. With the help of LCC sampling, our model generalizes well to unseen data rather than simply memorizing the training samples. These results demonstrate the effectiveness of the proposed sampling method in exploiting the local information of data.

TABLE 10  
Comparison of different GAN models equipped with and without the proposed LCC sampling method.

Resolution	DCGAN		StackGAN-v2		Progressive GAN	
	Baseline ( $d = 100$ )	with LCC-v2 ( $d = 30$ )	Baseline ( $d = 100$ )	with LCC-v2 ( $d = 30$ )	Baseline ( $d = 100$ )	with LCC-v2 ( $d = 30$ )
128×128						
256×256						

TABLE 11  
Effect of  $d_B$  and  $M$  on the performance of LCCGAN on Oxford-102.

Methods	Inception score	FID	Visual results
with entire bases	$3.314 \pm 0.043$	63.2	
with local bases	$3.370 \pm 0.031$	57.7	

## 7.2 Latent Manifold Interpolations

In this experiment, we conduct latent manifold interpolations on the Oxford-102 dataset. Specifically, given the generated image 1 and image 2 (See the first and the last column of Table 9, we have two corresponding LCC codings, and then interpolate several codings between these two LCC codings. Taking these interpolated codings as inputs, LCCGAN is able to interpolate a series of images between the generated image 1 and image 2. From Table 9, LCCGAN is able to interpolate realistic and smooth generated images.

## 7.3 Comparisons between Local and Entire Bases

In this experiment, we implement LCCGAN by directly sampling random  $\gamma$  and computing  $\nabla\gamma$ . However, using the entire bases would sample meaningless points to generate images with poor quality. In contrast, LCCGAN using local bases is able to exploit local information to improve the quality of generated images. To verify this, we conduct experiments to demonstrate the effectiveness of the sampling using local bases. From Table 11, LCCGAN with local bases has the largest IS and the lowest FID, and thus generates the most realistic images (as shown in the last column).

## 7.4 Impact of End-to-end Training Method

As shown in Algorithm 1, we adopt a multiple-stage method to train the LCCGAN models. Actually, similar to ALI [48] and BiGAN [49], we can also train LCCGAN models in an end-to-end manner. Specifically, we optimize a joint objective function by combining the loss of autoencoder, the objective of LCC, and the objective of GAN. However, the end-to-end training method may obtain inaccurate bases since it has to compensate for the

TABLE 12  
Impact on end-to-end training of LCCGAN on Oxford-102.

Methods	Inception score	FID	Visual results
end-to-end	$3.140 \pm 0.038$	65.5	
multiple-stage	$3.370 \pm 0.031$	57.7	

objectives of autoencoder and GAN. With such inaccurate bases, the performance of LCCGAN would deteriorate. To verify this, we conduct an experiment to compare the end-to-end with our multiple-stage training method. We show the results in Table 12. From these results, the model with the multiple-stage strategy significantly outperforms the model with end-to-end manner in terms of IS, FID, and visual results.

## 7.5 Comparison of High-resolution Images

The proposed LCCGAN has good scalability to generate high-resolution images. To verify this, we conduct experiments for the generation of high-resolution images, and the results are shown in Table 10. Specifically, since the proposed LCCGAN is a general GAN framework, we can apply the LCC learning method to a variety of GAN models, such as DCGAN, StackGAN-v2, and Progressive GAN. In this experiment, we compare the performance of different GAN models equipped with and without LCC sampling in Table 10. From these results, with a low input dimension  $d = 30$ , the models with the LCCGAN framework is able to generate more photo-realistic high-resolution images than the baseline models with  $d = 100$  under the resolutions of  $128 \times 128$  and  $256 \times 256$ .

## 7.6 Ablation Studies

In this experiment, we conduct many ablation studies on different hyperparameters and discuss the ratio of the number of class of data and the number of local bases.

**Impact of Hyperparameters  $L_v$  and  $L_h$**  In this experiment, we investigate the impact of  $L_v$  and  $L_h$  on the performance of the proposed method. In Table 15, we compare the performance with

TABLE 13  
Ablation study on  $d_B$  and  $M$  in terms of IS and FID on Oxford-102. We set  $d = 30$  for all the experiments.

Methods	Setting $M = 128$								Setting $d_B = 100$							
	$d_B = 50$		$d_B = 100$		$d_B = 200$		$d_B = 400$		$M = 64$		$M = 128$		$M = 256$		$M = 512$	
	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID
LCCGAN-v1	2.895	131.5	3.003	61.9	3.104	66.4	3.246	61.6	2.937	99.3	3.003	61.9	3.148	94.0	3.152	93.6
LCCGAN-v2	2.673	124.8	<b>3.370</b>	<b>57.7</b>	3.362	62.8	3.276	62.0	3.131	66.3	<b>3.370</b>	<b>57.7</b>	3.068	76.8	3.211	63.3

TABLE 14  
Visual comparison of the images produced by the models trained with different  $d_B$  and  $M$  on Oxford-102.

Methods	Setting $M=128$				Setting $d_B=100$			
	$d_B=50$	$d_B=100$	$d_B=200$	$d_B=400$	$M=64$	$M=128$	$M=256$	$M=512$
LCCGAN-v1								
LCCGAN-v2								

TABLE 15

Discussion of  $L_h$  and  $L_\nu$  on Oxford-102 with  $d = 30$ . IS represents Inception Score and FID represents Fréchet Inception Distance.

$L_\nu$	0.0001	0.001	0.01	0.1	1	10
IS	<b>3.370</b>	2.817	2.247	2.949	2.296	2.035
FID	<b>57.7</b>	205.9	255.1	280.8	262.9	269.8
$L_h$	0.0001	0.001	0.01	0.1	1	10
IS	1.890	2.228	1.984	3.159	<b>3.370</b>	3.239
FID	247.9	286.8	240	77.1	<b>57.7</b>	58.0

different hyperparameters on Oxford-102 given a dimensional input  $d = 30$ . From Table 15, the performance deteriorates with the increase of  $L_\nu$ . In terms of  $L_h$ , we obtain the best performance with  $L_h = 1$ . Thus, we set  $L_\nu = 0.0001$  and  $L_h=1$  in practice.

**Ablation study on  $d_B$  and  $M$ .** In this experiment, we conduct ablation studies on the Oxford-102 dataset to investigate how to choose the dimension of latent space ( $d_B$ ) and the number of bases ( $M$ ). Specifically, we study the impact of  $d_B$  by setting  $M=128$ , and similarly study the impact of  $M$  by setting  $d_B=100$ . From Table 13, when setting  $d_B=100$  and  $M=128$ , both LCCGAN-v1 and LCCGAN-v2 yield significantly better performance than the settings with a small  $d_B=50$  or a small  $M=64$ . If we further increase  $d_B$  and  $M$ , it would introduce additional computational cost but does not yield significant performance improvement. Therefore, in practice, we set the dimension of latent space and the number of bases as  $d_B=100$  and  $M=128$ , respectively. Furthermore, we also provide visual comparison of the images produced by the models trained with different  $d_B$  and  $M$  in Table 14. From the results, LCCGAN is able to generate images with promising quality when  $d_B=100$  and  $M=128$ .

**Discussions on the ratio of #class to  $d$ .** In this experiment, we investigate the ratio of the number of classes (#class) to the number of local bases  $d$ . To this end, we fix  $d$  to study the impact of the number of classes by varying #class on Oxford-102 (containing 102 classes). Note that with the increase of #class, the number of training samples will increase, which, however, would affect the performance of GANs. To be specific, according

to Theorem 2, the more training samples  $N$  we have, the better generalization performance of GANs would obtain. To remove the influence of the number of training samples, we sample images from different classes and keep the total number of training samples fixed. Specifically, we set  $N$  to be the smallest number of training samples in the case of #class=30, i.e.,  $N=1739$ . By setting  $d=3$  and  $d=5$ , we show the results in terms of IS and FID in Table 16. From these results, when we increase the number of classes from 30 to 102, the data become more complicated and thus need more local bases to represent the manifold of data. As a result, given a fixed number of local bases  $d$ , the images generated by the LCCGAN-v2 models tend to yield worse performance with the increase of #class.

TABLE 16  
Impact of #class on the performance of LCCGAN-v2 models in terms of IS and FID.

Input dimension	#class=30		#class=50		#class=70		#class=102	
	IS	FID	IS	FID	IS	FID	IS	FID
$d = 3$	3.087	104.8	2.902	119.9	2.742	125.8	2.697	130.1
$d = 5$	3.116	111.3	2.881	125.3	2.655	140.1	2.500	167.6

## 7.7 Comparisons in terms of Intra-FID

To verify the diversity and quality of the generated data, we train a single GAN model over different classes and evaluate the method using the intra-FID [50]. Such a metric first computes an FID score separately for each condition/class and then reports the average score over all conditions. However, in this paper, we focus on unconditional GAN methods and they have no conditions/labels associated with the generated images. As a result, we cannot directly compute the intra-FID for our method. To address this, we first train a classification model to classify the generated images into different classes, and then obtain the intra-FID score by computing an FID score for each class.

In this experiment, we train all GAN models over two LSUN classes (i.e., LSUN-classroom and LSUN-bedroom) and the classification model becomes a binary model (with the average

accuracy of 95.1%). We report both FID score for each class and the intra-FID scores of different methods in Table 17. From these results, our LCCGAN yields the smallest intra-FID among all the considered methods. It means that LCCGAN-v1 and LCCGAN-v2 are able to generate diverse samples by exploiting the local coordinate coding to capture the local information of data for each class. Moreover, LCCGAN-v2 achieves better performance than LCCGAN-v1 with the same dimension of inputs because LCCGAN-v2 has better approximation of generative models. In contrast, other GAN methods have poor diversity especially when the dimension is low since they using global coordinate coding may sample meaningless points.

TABLE 17

Comparisons with different GAN methods in terms of intra-FID on LSUN. We set  $d = 30$  for all the experiments.

Methods	LSUN-classroom (FID)	LSUN-bedroom (FID)	Intra-FID
DCGAN	182.38	212.82	197.60
StackGAN-v2	162.05	134.22	148.14
Progressive GAN	178.17	165.01	171.59
LCCGAN-v1	107.19	94.89	101.04
LCCGAN-v2	<b>97.87</b>	<b>90.58</b>	<b>94.23</b>

## 8 CONCLUSION

We have proposed a novel generative model by exploiting the local information on the latent manifold of real data to improve using local coordinate coding (LCC). Unlike existing methods, we develop an LCC-based sampling method to exploit the local information on the latent manifold of real data. Moreover, we also propose an advanced LCCGAN-v2 by introducing a higher-order term in the generator approximation. With LCC sampling, we can theoretically prove that a small-dimensional input is able to achieve good performance. Qualitative and quantitative experiments on several benchmark datasets demonstrate the effectiveness of the proposed method over several baseline methods.

## ACKNOWLEDGMENTS

This work was partially supported by Key-Area Research and Development Program of Guangdong Province (2018B010107001, 2019B010155002, 2019B010155001), National Natural Science Foundation of China (NSFC) 61836003 (key project), 2017ZT07X183, Tencent AI Lab Rhino-Bird Focused Research Program (No.JR201902), Fundamental Research Funds for the Central Universities D2191240.

## REFERENCES

- [1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, 2014.
- [2] J. Cao, Y. Guo, Q. Wu, C. Shen, and M. Tan, “Adversarial learning with local coordinate coding,” in *Proceedings of the International Conference on Machine Learning*, 2018.
- [3] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *Proceedings of the International Conference on Machine Learning*, 2017.
- [4] M. Zhu, P. Pan, W. Chen, and Y. Yang, “Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.
- [5] J. Lin, Z. Chen, Y. Xia, S. Liu, T. Qin, and J. Luo, “Exploring explicit domain supervision for latent space disentanglement in unpaired image-to-image translation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2019.
- [6] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, “On the effectiveness of least squares generative adversarial networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, pp. 2947–2960, 2019.
- [7] N. OTBERDOU, M. Daoudi, A. Kacem, L. Ballihi, and S. Berretti, “Dynamic facial expression generation on hilbert hypersphere with conditional wasserstein generative adversarial nets,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.
- [8] J. Pan, J. Dong, Y. Liu, J. Zhang, J. Ren, J. Tang, Y. W. Tai, and M. Yang, “Physics-based generative adversarial models for image restoration and beyond,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.
- [9] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra, “Video (language) Modeling: a Baseline for Generative Models of Natural Videos,” *arXiv preprint arXiv:1412.6604*, 2014.
- [10] M. Mathieu, C. Couprie, and Y. LeCun, “Deep Multi-scale Video Prediction beyond Mean Square Error,” in *International Conference on Learning Representations*, 2016.
- [11] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *IEEE International Conference on Computer Vision*, 2017.
- [12] T. Kim, M. Cha, H. Kim, J. Lee, and J. Kim, “Learning to discover cross-domain relations with generative adversarial networks,” in *Proceedings of the International Conference on Machine Learning*, 2017.
- [13] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, “Adversarial discriminative domain adaptation,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [14] F. Zhu, L. Zhu, and Y. Yang, “Sim-real joint reinforcement transfer for 3d indoor navigation,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.
- [15] S. Xie, Z. Zheng, L. Chen, and C. Chen, “Learning semantic representations for unsupervised domain adaptation,” in *Proceedings of the International Conference on Machine Learning*, 2018.
- [16] Y. Guo, Q. Chen, J. Chen, J. Huang, Y. Xu, J. Cao, P. Zhao, and M. Tan, “Dual reconstruction nets for image super-resolution with gradient sensitive loss,” *arXiv preprint arXiv:1809.07099*, 2018.
- [17] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.
- [18] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” in *International Conference on Learning Representations*, 2018.
- [19] Y. Guo, Q. Chen, J. Chen, Q. Wu, Q. Shi, and M. Tan, “Auto-embedding generative adversarial networks for high resolution image synthesis,” *arXiv preprint arXiv:1903.11250*, 2019.
- [20] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, 2006.
- [21] J. B. Tenenbaum, V. De Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, 2000.
- [22] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, 2000.
- [23] K. Yu, T. Zhang, and Y. Gong, “Nonlinear learning using local coordinate coding,” in *Advances in Neural Information Processing Systems*, 2009.
- [24] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” in *Advances in Neural Information Processing Systems*, 2017.
- [25] G.-J. Qi, L. Zhang, H. Hu, M. Edraki, J. Wang, and X.-S. Hua, “Global versus localized generative adversarial nets,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [26] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [27] I. Tolstikhin, B. Olivier, S. Gelly, and B. Schoelkopf, “Wasserstein auto-encoders,” in *International Conference on Learning Representations*, 2018.
- [28] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, “Adversarial autoencoders,” *arXiv preprint arXiv:1511.05644*, 2015.
- [29] G. K. Dziugaite, D. M. Roy, and Z. Ghahramani, “Training generative neural networks via maximum mean discrepancy optimization,” in *Uncertainty in Artificial Intelligence*, 2015.
- [30] H. Thanh-Tung, T. Tran, and S. Venkatesh, “Improving generalization and stability of generative adversarial networks,” in *International Conference on Learning Representations*, 2019.
- [31] H. Jiang, Z. Chen, M. Chen, F. Liu, D. Wang, and T. Zhao, “On computation and generalization of generative adversarial networks under spectrum control,” in *International Conference on Learning Representations*, 2019.

- [32] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang, "Generalization and equilibrium in generative adversarial nets (GANs)," in *Proceedings of the International Conference on Machine Learning*, 2017.
- [33] P. Zhang, Q. Liu, D. Zhou, T. Xu, and X. He, "On the discrimination-generalization tradeoff in GANs," in *International Conference on Learning Representations*, 2018.
- [34] J. Cao, L. Mo, Y. Zhang, K. Jia, C. Shen, and M. Tan, "Multi-marginal wasserstein gan," in *Advances in Neural Information Processing Systems*, 2019.
- [35] K. Yu and T. Zhang, "Improved local coordinate coding using local tangents," in *Proceedings of the International Conference on Machine Learning*, 2010.
- [36] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, 2002.
- [37] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.
- [38] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.
- [39] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.
- [40] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *IEEE International Conference on Computer Vision*, 2015.
- [41] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Adversarial generator-encoder networks," *arXiv preprint arXiv:1704.02304*, 2017.
- [42] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *IEEE International Conference on Computer Vision*, 2017.
- [43] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, 2016.
- [44] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017.
- [45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.
- [46] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *IEEE International Conference on Computer Vision*, 2015.
- [47] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan++: Realistic image synthesis with stacked generative adversarial networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1947–1962, 2018.
- [48] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," in *International Conference on Learning Representations*, 2017.
- [49] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," in *International Conference on Learning Representations*, 2016.
- [50] T. Miyato and M. Koyama, "cGANs with projection discriminator," in *International Conference on Learning Representations*, 2018.
- [51] F. Bach, "Breaking the curse of dimensionality with convex neural networks," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 629–681, 2017.
- [52] M. Ledoux and M. Talagrand, *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media, 2013.
- [53] D. J. Hsu, S. M. Kakade, J. Langford, and T. Zhang, "Multi-label prediction via compressed sensing," in *Advances in Neural information processing systems*, 2009, pp. 772–780.



**Jiezhang Cao** is a Master student in the School of Software Engineering at South China University of Technology. He received his bachelor's degree in Statistics from Guangdong University of Technology in 2017. His research interests include theoretical machine learning and computer vision.



**Yong Guo** is a Ph.D. student in the School of Software Engineering at South China University of Technology. He received his bachelor's degree in Software Engineering from the same university in 2016. His research interests include deep learning and computer vision.



**Qingyao Wu** received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2013. He was a Post doctoral Research Fellow with the School of Computer Engineering, Nanyang Technological University, Singapore, from 2014 to 2015. He is currently a Professor with the School of Software Engineering, South China University of Technology, Guangzhou, China. His current research interests include machine learning, data mining, big data research.

**Chunhua Shen** is a Professor of Computer Science at The University of Adelaide, Australia.



**Junzhou Huang** received the BE degree from the Huazhong University of Science and Technology, China, the MS degree from the Chinese Academy of Sciences, China, and the PhD degree from Rutgers University. He is an associate professor with the Computer Science and Engineering Department, University of Texas at Arlington. His major research interests include machine learning, computer vision, and imaging informatics. He was selected as one of the 10 emerging leaders in multimedia and signal processing by the IBM T.J. Watson Research Center in 2010. He received the NSF CAREER Award in 2016.



**Mingkui Tan** is currently a professor with the School of Software Engineering at South China University of Technology. He received his Bachelor Degree in Environmental Science and Engineering in 2006 and Master degree in Control Science and Engineering in 2009, both from Hunan University in Changsha, China. He received the Ph.D. degree in Computer Science from Nanyang Technological University, Singapore, in 2014. From 2014-2016, he worked as a Senior Research Associate on computer vision in the School of Computer Science, University of Adelaide, Australia. His research interests include machine learning, sparse analysis, deep learning and large-scale optimization.