Geometric View of GAN and Visualization

Zhizhong Li
The Chinese University of Hong Kong
lz015@ie.cuhk.edu.hk

Xiao Chu, Xiaogang Wang
The Chinese University of Hong Kong
{xchu,xgwang}@ee.cuhk.edu.hk

Abstract

We view the task of Generative Adversarial Networks as manifold learning. Instead calling it as noise, we see the latent space as the coordinate of the data manifold. The generator is the function that maps coordinates to data manifold. Thus, other than the traditional approach that investigating the probabilistic properties of the noise distribution and the data distribution, we ask whether the geometric properties of latent space and data manifold interact with each other. Specifically, we visualize the effect of dimensionality and connectivity of latent space and data manifold using specially designed synthetic experiments.

1 Introduction

Generative Adversarial Networks (GAN) [7] as generative models have been actively studied and developed [2–6, 9, 10, 13–16, 19, 22–25] in the last few years. There are theoretical discussions [2, 15, 23], various extensions [3, 4, 6, 9, 13, 14, 19], exploring effective network design and training methods [1], and applications in image generation [5, 16, 22], manipulation [18, 24], and cross domain transfer [10, 25], etc. Compared to Restricted Boltzmann Machines [8], Variational Auto-Encoders, or other generative models, GAN is able to generate higher quality examples [11] However, GAN also faces some well known problems, such as hard to train, mode collapse, and lack of systematic evaluation methods, to name a few.

GAN is formulated as a two-player game that involves a generator G and a discriminator D. Given a data distribution that we want to model, the generator is trained to generate samples that look real, while the discriminator is trained to distinguish between samples that come from real data distribution and those come from the generator. It can be shown [7] that if in each step, the discriminator is at its optimum, then the objective of GAN is equivalent to minimizing the Jensen-Shannon divergence between the real data distribution and the generated sample distribution. At the equilibrium state, the discriminator cannot identify the source of a sample, and the generator is able to generate samples that share the same distribution as the real data.

It is also natural to interpret GAN in the context of manifold learning. In real applications, we are dealing with structured data such as natural images. The distribution of such data often concentrated on a low dimensional manifold. So we can view the target of GAN is to learn such a data manifold which is parametrized by the latent space. The authors of Wasserstein GAN [1, 2] adopt this interpretation to explain why GANs are so hard to train. Intuitively speaking, both the data manifold and the generated manifold are low dimensional manifolds in a high dimensional ambient space, which means that they almost never have sufficient overlap. In this case, the Jensen Shannon divergence will have trouble by definition and this accounts for the difficulties in training. This evidenced that the geometric view is helpful in understanding GAN.

Since this close relationship between geometry and GAN, we are inspired to investigate more deep into the geometric aspects of GAN. There are many important concepts for a manifold, for example the dimensionality and the connectivity. All these aspects lack sufficient discussion in current literature. In this report, we study the effect of these geometric properties on the training of GAN.

Paper Report of Final Project for ELEG5491 [20] - Introduction to Deep Learning.

1.1 Related Work

The geometric viewpoint appears in many works on GAN [1–3, 23–25]. As discussed above, it explains the difficulties encountered in training GAN [1]. WGAN [2] exploits the good behavior of the Wasserstein distance in measuring two distributions that are concentrated on low-dimensional manifolds. In [21], the author try to convince us that a spherical latent space is better than a cube. Notice the topological difference between cube and sphere in this case. Mode regularized GAN [3] introduces mode regularizer and manifold-diffusion training based on the geometric interpretation. The geometric viewpoint is also implicitly referred to in many works through the casual usage of terminologies like manifold. The benefit is to provide a clear intuition that facilitate understandings.

2 Geometric View of GAN

A *manifold* (without boundary) is a set of points that locally resembles Euclidean space near each point. Specifically, for each point of an n-dimensional manifold, there exists a neighborhood that is homeomorphic to an n-dimensional open set $\Omega \subset \mathbb{R}^n$, such as the open cube $\mathring{I}^n := (0,1)^n \subset \mathbb{R}^n$. Intuitively, we can view a n-dimensional manifold as a *surface* in an Euclidean space. Trivial examples are, a *point* in a line, a *circle* in a plane, or a *ball* in the 3-dimensional space we live in.

Suppose we have an n-dimensional data manifold M in the N-dimensional ($N \geq n$) Euclidean space. For simplicity, we assume M is *contractible*, *i.e.*, it is topologically equivalent to a point. Choose the cube $\Omega = \mathring{I}^n$ as the latent space, then we are able to parametrize manifold M by a single coordinate chart $(z_1, \ldots, z_n)$. Suppose the mapping from coordinate space to manifold is

$$\varphi : \Omega \to M \subset \mathbb{R}^N, \tag{1}$$

then we can view the objective of GAN as to learn such a parametrization as φ. There are several immediate observations on GAN from this geometric point of view.

Decouple Geometry and Distribution In fact, as a generative model, being able to generate the whole set of real data points is not enough. It should also generate samples in the right probability. Thus, it is natural to decouple the generation task as two subtasks: generating the right geometry, and the right distribution. Following this idea, we can first focus on generating the correct data manifold without worrying about issues on the distribution. For example, we can exploit sampling tricks more freely to aid this geometry learning process. Once the right geometry is generated, we can freeze the generator and attach another network right before the latent space to learn to adjust to the right distribution. The latter task would be easier since 1) the dimension of latent space is lower, and 2) the data manifold and generated manifold already have sufficient overlap.

Inverse Mapping The coordinate mapping φ is a homeomorphism, and thus a bijection. The necessary and sufficient condition for φ being a bijection is the existence of a mapping ψ,

$$\psi : M \to \Omega \subset \mathbb{R}^n, \tag{2}$$

such that,

$$\psi \circ \varphi = \mathrm{id}_\Omega \ (\varphi \text{ is injective}), \quad \text{and} \quad \varphi \circ \psi = \mathrm{id}_\mathcal{M} \ (\varphi \text{ is surjective}). \tag{3}$$

The mode collapse problem might be mitigated provided we impose the bijection as an regularization, because data samples are explicitly required to be generatable. In fact, some works [3, 9, 10, 18, 24] already take the general idea of inverse mapping into consideration. Though they mostly motivated from the auto-encoder perspective and focus more on the reconstruction, i.e., the second equation in (3). The full power of inverse mapping is waiting to be discovered.

Learning Mapping as Graph In conditional GAN, we want to control the generated sample x through some condition c. This can be reformulated as learning a multi-valued mapping from condition c to some data sample x. Geometrically, we can represent a mapping by its graph $G := \{(c, x)\}$. The graph is a manifold that can be learned in the usual GAN formulation. The idea of learning mapping as graph is a general solution to handle the one to many relationships. More applications other than conditional GAN are still underdeveloped.

These observations lead to some useful insights into GAN. As we have commented, some of the ideas have been explored recently in one form or another. However, many aspects are still in mystery.

In this report, we take a crack on the simplest yet a fundamental problem in understanding the geometric aspects of GAN, namely the impact of the geometric properties of latent space and data manifold on the training of GAN. Specifically, we are interested in the following two properties:

- *Dimensionality*. We know that two manifolds never match if their dimensions are different in the first place. However, in real applications, the dimension of data manifold is unknown. So it is interesting to see what would happen to GAN if the dimensions of latent space and data manifold do not match.
- *Connectivity*. Natural images are clustered into distinct classes. This means that the data manifold may have several disconnected components. The questions are, do we have to make the latent space disconnected at the same time? What if the latent space is disconnected while the data distribution is connected?

3 Experiments and Visualization

To answer the questions raised in previous section, we specially designed some learning problems in low-dimensions, so that we can visualize what happens in GAN. Both the latent space and the data space is restricted to 1-dimension or 2-dimension. We use *fully connected* (fc) layers as the building blocks for both the generator and the discriminator. A *batch normalization* layer and a *Elu* activation layer follow after each fc layer. Across all experiments, Both the generator and discriminator have $20-40-100-200-200-100-40-20$ hidden neuron numbers. The capacity of these networks is large enough for our experiments. We use *rmsprop* with initial lr 0.001, weight decay 0.0005, and batch size 256 as the optimization strategy. The learning rate is multiplied by 0.98 after every 100 iterations. Generator and discriminator are updated alternately, with each processing 1 batch.

We use *Parrots* [17] as the deep learning framework and use its Julia port *Parrots.jl* [12] as the working language. All codes were implemented from scratch based on the algorithm described in [7]. Codes and detailed results are available at `https://github.com/innerlee/ELEG5491`.

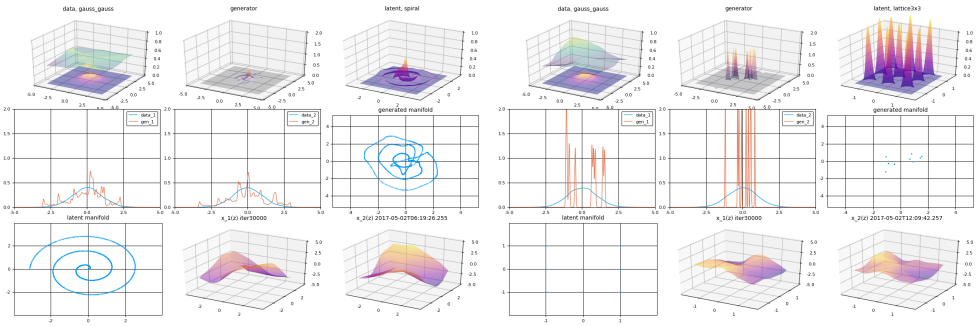
For the visualizations in Figure 1 and Figure 2, we draw nine subplots for each setting. They are 1) upper left: data distribution and discriminator values. 2) upper middle: generated distribution. 3) upper right: latent space distribution. 4) middle left: marginal distribution of generated and data distribution along first axis. 5) middle middle: marginal distribution of generated and data distribution along second axis. 6) middle right: generated manifold. 7) bottom left: latent manifold. 8) bottom middle, first component of generator function, x, y-axes are latent coordinates, z-axis is the function value. 9) bottom right, second component of generator function.

3.1 Dimensionality

We use three manifolds: 1) A 3×3 lattice. It is 0-dimensional manifold in $\mathbb{R}^2$. 2) A 1-dimensional spiral curve in $\mathbb{R}^2$. 3) the whole 2-dimensional plane with Gaussian distribution. As shown in Figure 1, we can see that 1) If the dimension of latent manifold is lower than that of the data manifold (Figure 1 (a, b)), the generated low dimensional manifold try to cover the larger dimensional data manifold as possible. 2) The marginal distribution of the generated manifold is close to that of the data manifold. 3) If the dimension of latent manifold is larger (Figure 1 (c, d)), then the generated manifold is able to cover the data manifold. However, it may generate many fake examples that do not belong to the data manifold. This is clearly shown in the Gaussian to spiral experiment.

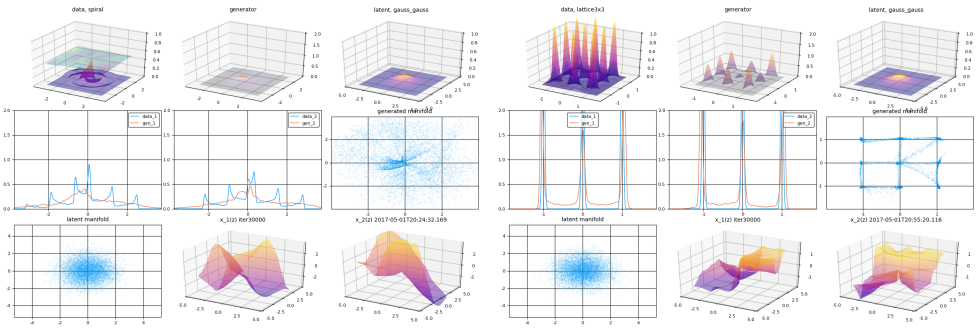
3.2 Connectivity

We have two settings: 1) 2-dimensional. Generating from a connected uniform square to a squared patches with 9 components, and vice versa. 2) 1-dimensional. Generating from a connected circle to four disconnected arcs, and vice versa. From Figure 2, we can observe that, 1) The connected latent space is able to approximate the disconnected data manifold (Figure 2 (a,c)). 2) The disconnected latent space has difficulty in covering the data manifold although the dimensions are the same (Figure 2 (b,d)). So a connected latent space is able to handle different components of a disconnected data manifold, and making the latent space disconnected is not good in general.



(a) z: 1-dim, spiral. x: 2-dim, Gaussian.

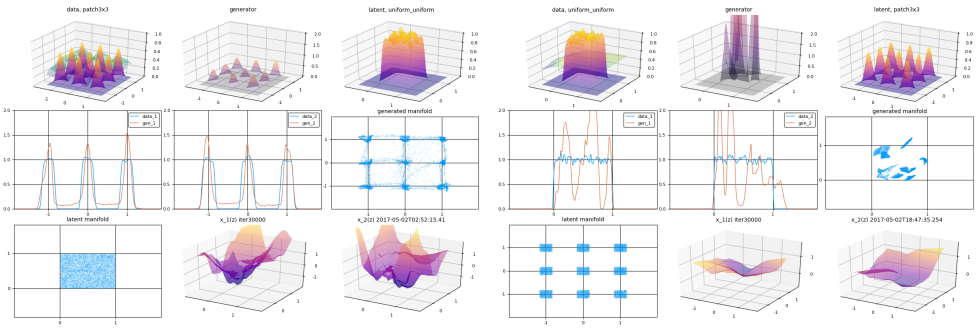
(b) z: 0-dim, lattice. x: 2-dim, Gaussian.



(c) z: 2-dim, Gaussian. x: 1-dim, spiral.

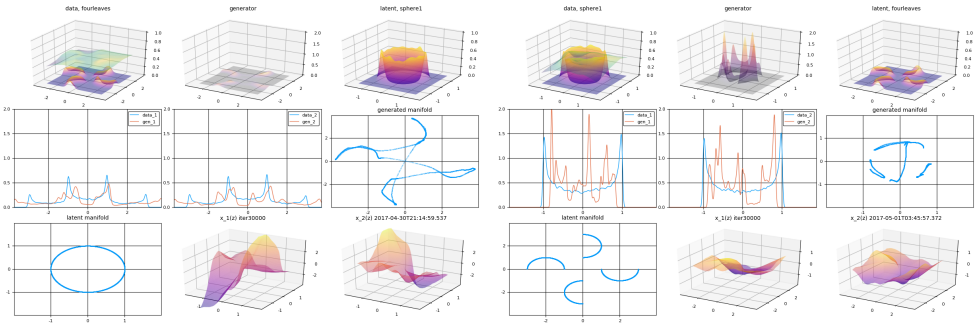
(d) z: 2-dim, Gaussian. x: 0-dim, lattice.

Figure 1: GAN in different dimensions. The snapshots are taken at iteration 30000.



(a) z: 2-dim, uniform square, connected. x: 2-dim, 9 square patches, disconnected.

(b) z: 2-dim, 9 square patches, disconnected. x: 2-dim, uniform square, connected.



(c) z: 1-dim, unit circle, connected. x: 1-dim, four arcs, disconnected.

(d) z: 1-dim, four arcs, disconnected. x: 1-dim, unit circle, connected.

Figure 2: GAN in connected/disconnected manifolds. The snapshots are taken at iteration 30000.

References

[1] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. In *NIPS 2016 Workshop on Adversarial Training. In review for ICLR*, 2017.

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[3] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016.

[4] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

[5] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015.

[6] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[7] 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*, pages 2672–2680, 2014.

[8] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[9] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie. Stacked generative adversarial networks. *arXiv preprint arXiv:1612.04357*, 2016.

[10] T. Kim, M. Cha, H. Kim, J. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*, 2017.

[11] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint arXiv:1609.04802*, 2016.

[12] Z. Li and D. Lin. Parrots.jl: Julia port of parrots. https://github.com/ParrotsDL/Parrots.jl. Accessed: 2017.

[13] L. Mescheder, S. Nowozin, and A. Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.

[14] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[15] S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pages 271–279, 2016.

[16] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.

[17] Parrots. Parrots deep learning platform. http://www.parrotsdnn.org/. Accessed: 2017.

[18] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016.

[19] 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*, pages 2226–2234, 2016.

[20] X. Wang. Introduction to deep learning. http://dl.ee.cuhk.edu.hk/. Accessed: 2017.

[21] T. White. Sampling generative networks: Notes on a few effective techniques. *arXiv preprint arXiv:1609.04468*, 2016.

[22] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016.

[23] J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.

[24] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.

[25] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.